```
Set     Items   Description
S1      20494   (DIGITAL? OR MULTIMEDIA?)()(GOOD? OR PRODUCT? OR ENTIT? OR
             MODULE? OR UNIT? OR DEVICE?) OR MPEG? ? OR MP3? ?
S2     771034   SOFTWARE? OR NONSOFTWARE? OR DVD? ? OR CDROM? OR CD()ROM? ?
              OR DISK? OR DISC? ? OR FLOPPY? OR FLOPPIE?
S3      14439   (AUDIO?(10N)VIDEO?)(2N)(DATA? OR GOOD? OR PRODUCT? OR MODU-
             LE? OR ENTIT? OR UNIT? OR DEVICE?)
S4        218   SBOX? OR S()(BOX OR BOXES)
S5          0   RIJNDAELSBOX? OR S()FUNCTION?()(BOX OR BOXES)
S6          1   (SOFTWARE? OR SOFT()WARE?)()USAGE?()MONITOR?()(BOX OR BOXE-
             S) OR SUBSTITUT?()FUNCTION?()(BOX OR BOXES)
S7         27   SUBSTITUTION?()(LOOKUP OR LOOK?()UP)()TABLE? OR SUBSTITUT?-
             ()LUT? ? OR SUBSTITUT?()(BOX OR BOXES)
S8          0   RANDOM?()SUBSTITUT?()(LUT? ? OR BOX OR BOXES OR TABLE?)
S9    4826429   PARTITION? OR PART? ? OR PARTIAL? OR SEGMENT? OR DIVISION?
S10   1333872   PARCEL? OR PIECE? OR CHUNK? OR FRACTION? OR SLICE? OR DIVI-
             D?
S11   3501297   SECTION? OR SECTOR? OR PORTION? OR APPORTION? OR SECTOR?
S12   3951607   FIRST? OR 1ST OR PRIMARY OR INITIAL? OR ORIGINAL? OR LEADO-
             FF? OR MAIN OR CHIEF OR INTRODUCTORY? OR MASTER?
S13    322126   SUBSTITUT? OR PROXIE? OR PROXY? OR STANDIN OR STANDINS OR -
             STAND?()IN
S14    867640   PSEUD? OR SYNTHE? OR ARTIFICIAL? OR TEMPORAR?
S15   5173224   SECOND? OR 2ND OR DOUBL? OR TWIN? OR EXTRA? OR ANOTHER OR -
             SUBSIDIAR? OR AUXILIAR? OR DIFFERENT? OR ALTERNAT? OR SLAVE?
S16     43173   ENCRYPT? OR ENCIPHER? OR ENCYPHER? OR SCRAMBL? OR HASH? OR
             CRYPT? OR ENSCRAMBL?
S17     13664   DECRYPT? OR DECIPHER? OR DECYPHER? OR DESCRAMBL? OR DEHASH?
              OR UNSCRAMBL? OR UNENCRYPT?
S18    413630   IC=(H04K? OR H04L?)
S19   1738712   MC=(T01? OR W01? OR W04?)
S20        50   S4:S8 AND S9:S12 AND S16:S17
S21        47   S20 AND (S1:S3 OR S12:S15 OR S18:S19)
S22         6   S1:S3 AND S4:S8 AND S9:S11
S23       104   S4:S8 AND S9:S11
S24        74   S23 AND (S1:S3 OR S12:S19)
S25       119   S20:S24
S26    848684   PR=2001:2006
S27       109   S25 NOT S26
S28       109   IDPAT (sorted in duplicate/non-duplicate order)
File 347:JAPIO Nov 1976-2005/Aug(Updated 051205)
         (c) 2005 JPO & JAPIO
File 350:Derwent WPIX 1963-2006/UD,UM &UP=200604
         (c) 2006  Thomson Derwent
```

28/3,K/13 (Item 13 from file: 350)
DIALOG(R)File 350:Derwent WPIX

012267330
WPI Acc No: 1999-073436/199907
XRPX Acc No: N99-053875
Block cipher secure against differential **and linear** cryptanalysis - divides **the input into two half-blocks which are combined with the key octet by octet, then shifted left after passing through** substitution boxes
Patent Assignee: SAMSUNG ELECTRONICS CO LTD (SMSU)
Inventor: CHA Y; LEE C; CHA Y T; LEE C H
Number of Countries: 006 Number of Patents: 010
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
FR 2765056	A1	19981224	FR 987753	A	19980619	199907	B
GB 2327581	A	19990127	GB 9811900	A	19980604	199907	
DE 19827904	A1	19990114	DE 1027904	A	19980623	199908	
JP 11073101	A	19990316	JP 98175844	A	19980623	199921	
GB 2327581	B	19990804	GB 9811900	A	19980604	199933	
KR 99002840	A	19990115	KR 9726558	A	19970623	200011	
DE 19827904	C2	20000511	DE 1027904	A	19980623	200028	
JP 3148181	B2	20010319	JP 98175844	A	19980623	200125	
US 6314186	B1	20011106	US 9895845	A	19980611	200170	
KR 389902	B	20030922	KR 9726558	A	19970623	200416	

Priority Applications (No Type Date): KR 9726558 A 19970623
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
FR 2765056	A1		20	H04L-009/28	
GB 2327581	A			H04L-009/06	
DE 19827904	A1			H04L-009/06	
JP 11073101	A		8	G09C-001/00	
GB 2327581	B			H04L-009/06	
KR 99002840	A			G06F-001/00	
DE 19827904	C2			H04L-009/06	
JP 3148181	B2		11	G09C-001/00	Previous Publ. patent JP 11073101
US 6314186	B1			H04L-009/28	
KR 389902	B			G06F-001/00	Previous Publ. patent KR 99002840

Block cipher secure against differential **and linear** cryptanalysis - ...

... divides **the input into two half-blocks which are combined with the key octet by octet, then shifted left after passing through** substitution boxes
...Abstract (Basic): The **encryption** algorithm **divides** the data stream into blocks of 2 N octets, and the blocks are **divided** into a **first** and a **second** half. An exclusive-OR operation is performed between the **second** half and a rotation key of M octets. The result of this step is **divided** into L blocks of eight bits, and the **first** block is sent to a **first S box** , and each of the remaining blocks sent to a corresponding **S - box** after it has been combined with the output of the preceding **S - box** .
...

...The output of each of the **S - boxes** is rotated left, and the results used to form a new **second** half of the input block, while the old **second** half forms a new half...

...USE - USE - **Encryption** of digital audio streams...

...ADVANTAGE - ADVANTAGE - Allows construction of **encryption** algorithm from blocks of fast algorithms to give fast **encryption** and **decryption** with algorithm that is resistant to **differential** and linear **cryptanalysis** .

...Title Terms: **DIFFERENTIAL** ;
...International Patent Class (Main): **H04L-009/06** ...

... **H04L-009/28**
Manual Codes (EPI/S-X): **T01-E02** ...

... **T01-J04** ...

... **W01-A05A**

US006314186B1

(12) **United States Patent**
Lee et al.

(10) **Patent No.: US 6,314,186 B1**
(45) **Date of Patent: Nov. 6, 2001**

(54) **BLOCK CIPHER ALGORITHM HAVING A ROBUST SECURITY AGAINST DIFFERENTIAL CRYPTANALYSIS, LINEAR CRYPTANALYSIS AND HIGHER-ORDER DIFFERENTIAL CRYPTANALYSIS**

(75) Inventors: **Chang-hyi Lee**, Guachun; **Young-tae Cha**, Sungnam, both of (KR)

(73) Assignee: **Samsung Electronics Co., Ltd.** (KR)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/095,845**

(22) Filed: **Jun. 11, 1998**

(30) **Foreign Application Priority Data**

Jun. 23, 1997 (KR) 97-026558

(51) **Int. Cl.**[7] **H04L 9/28**
(52) **U.S. Cl.** **380/28**
(58) **Field of Search** 380/28

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,003,597	*	3/1991	Merkle	380/37
5,623,548	*	4/1997	Akiyama et al.	380/28
5,838,794		11/1998	Mittenthal	380/28

FOREIGN PATENT DOCUMENTS

WO 98/00949 1/1998 (WO) .

OTHER PUBLICATIONS

Schneier, Applied Cryptography, pp. 272, 320–323, 1996.*
Mittenthal, Statistical Efficient Inter-Round Mixing in Block Substitution Devices, Teledyne, 1996.*
Lee, et al., "The Block Cipher: Snake with Provable Resistance Against DC and LC Attacks," *Proc. of JW–ISC*, 1997, Session 1, pp. 3–17.
Knudsen, "Truncated and Higher Order Differentials," *Advances in Cryptology, Fast Software Encryption*, 1995.
Matsui, "Linear Cryptanalysis Method for DES Cipher," *Advances In Cryptology, Endocrypt*, 1993, pp. 391–397.
Nyberg, "S–Boxes and Round Functions with Controllable Linearity and Differential Uniformity," *Advances In Cryptology, Fast Software Encryption*, 1994, pp. 111–129.
Internet thread concerning Teledyne, cypherpunks@toad.com, Jul. 1996.*

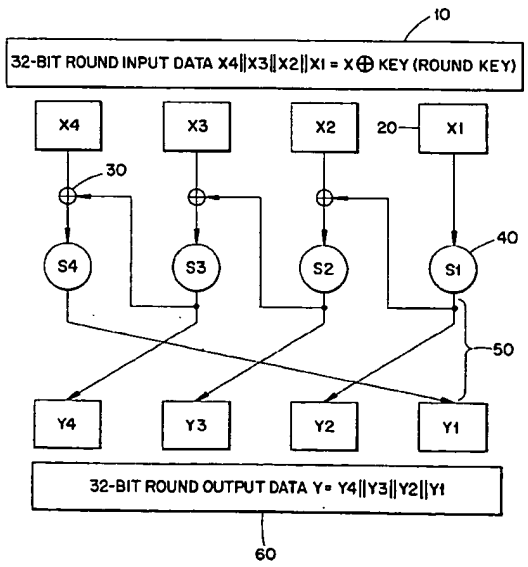
* cited by examiner

Primary Examiner—Gail Hayes
Assistant Examiner—James Seal
(74) *Attorney, Agent, or Firm*—Burns, Doane, Swecker & Mathis, L.L.P.

(57) **ABSTRACT**

The present invention relates to the block cipher algorithm based on the prior Feistel type block cipher algorithm (or similar to DES algorithm). Usually the security of Feistel type block cipher algorithm depends on the structure of its round function. More specifically, the present invention relates to the round function structure of the Feistel type block cipher algorithm, in the instance that the round input data block is divided into 8-bit blocks and the divided sub-blocks are fed, with the combined output data of the previous S-box, into 256x8 S-box, except for the first input sub-data block. The first sub-data block one is directly fed into the first S-box. The total output data block, after these steps, is rotated by 8-bits and this rotated result is the output of the current round function.

26 Claims, 4 Drawing Sheets



Block **180** illustrates the 32-bit right half of the 64-bit input plain text to the cipher SNAKE. Block **190** illustrates the round function of SNAKE of which structure was described in FIG. **1**. Block **200** illustrates a round key generated from SNAKE's key scheduling process shown in FIGS. **2A–2B**. Block **210** illustrates a logic exclusive-OR operator. Block **220** illustrates the 32-bit left half of the 64-bit final output data such as encrypted data or cipher text, through the 16 round process of SNAKE which executes a one-round process, repeated 16 times. Block **230** illustrates the 32-bit right half of the 64-bit final output data such as encrypted data or cipher text through the 16 round process of SNAKE which executes a one-round process, repeated 16 times.

In summary, SNAKE operates on a 64-bit block of plain text. The block is broken into a right half and a left half, each 32-bits long. Then there are 16 rounds of identical operations, with reference to FIG. **3**, called round function F, in which the data are combined with the key via XOR-operation. After the sixteenth round, the right and left halves are joined, and the algorithm is completed. In each round the right half (32-bits) of the previous round's output data is combined with its round key (via XOR) and the resulting data is broken into four 8-bit data blocks, X1, X2, X3, X4.

The data blocks form the input data to the previously described round function F. Again, this output data of F are combined with the 32-bit left half data block via XOR to be the next or new round's right half data block (the old right half becomes the new left half). These operations are repeated 16 times, thereby making 16 rounds of SNAKE.

If B_j is the result of the j-th iteration, L_j and R_j are the left and right halves of B_j, K_j is the key for the round j, and F is the round function described previously, then a round looks like:

$L_j=R_{j-1}$;

$R_j=L_{j-1}\oplus F(R_{j-1}\oplus K_j)$;

In the present invention, the security (resistance) of SNAKE could be deduced from consideration of the output difference data from each S-box is seen if given a pair of data values with a difference (input difference) as variable parameters, and constructing a linear system of equations of the difference variable parameters to get its coefficient matrix, which is called 'Transient Differential Matrix'. By making or finding some conditions to confine the cipher to its 'rank', the round function structure of SNAKE can be deduced. The proof of the security was disclosed in Changhyi Lee and Young-tae Cha, "The Block Cipher: SNAKE with Provable Resistance against-DC and LC Attacks", JW-ISC, (1997), herein incorporated by reference.

The processing speed of the present invention is faster than that of DES. In the simulation of SNAKE, implemented in the C++ language on a 120 MHz PENTIUM PC, the encryption process of the present invention performs at 16 Mbps, while DES performs at 10.4 Mbps on the same machine.

The invention may be embodied in a general purpose digital computer that is running a program or program segments originating from a computer readable or usable medium, such medium including, but not limited to, magnetic storage media (e.g., ROMs, floppy disks, hard disks, etc.), optically readable media (e.g., CD-ROMs, DVDs, etc.), and carrier waves (e.g., transmissions over the Internet). A functional program, code and code segments, used to implement the present invention can be derived by a skilled computer programmer from the description of the invention contained herein.

The previous description of the exemplary embodiments is provided to enable any person skilled in the art to make or use the present invention. The various modifications to these embodiments will be readily apparent to those skilled in the art, and the generic principles defined herein may be applied to other embodiments without the use of the inventive faculty. Thus, the present invention is not intended to be limited to the embodiments shown herein but is to be accorded the widest scope consistent with the principles and novel features disclosed herein.

What is claimed is:

1. A block cipher method having a round process and having a key scheduling algorithm, comprising:

(a) dividing a data stream into 2N-byte data blocks, each block being divided into a first half block and a second half block;

(b) executing a logical exclusive-OR operation with the second half block and an N-byte round key;

(c) dividing a result of step (b) into N divided blocks, sending a first divided block to a first S-box S1 and sending to each remaining S-box S2, . . . , Sn a result of executing a logical exclusive-OR operation of each corresponding divided block with output data from the previous S-box;

(d) rotating an N-byte result of step (c) to the left by M bits;

(e) executing a logical exclusive-OR operation with the first half block and a result of step (d);

(f) relabeling the second half block as a new first half block for use in a next round, said next round utilizing a next round key;

(g) relabeling a result of step (e) as a new second half block for use in said next round;

(h) executing subsequent rounds by repeating steps (b) through (g) until just before a final round;

(i) sending a final second half block to a right half of a final output and executing a logical exclusive-OR operation with the final second half block and a final N-byte round key;

(j) dividing a result of the logical exclusive-OR operation of step (i) into N final-round blocks, sending a first final-round block to the first S-box and sending to each remaining S-box a result of executing a logical exclusive-OR operation of each corresponding final-round block with output data from the previous S-box;

(k) rotating an N-byte result of step (j) to the left by M bits; and

(l) sending a result of executing a logical exclusive-OR operation with a final first half block and a result of step (k) to a left half of the final output.

2. The block cipher method of claim **1**, wherein said round keys are generated by a method comprising the steps of:

(m) breaking N-byte seed key data into N seed sub-blocks;

(n) executing modular addition with a first seed sub-block and an N-th seed sub-block and sending a result of this modular addition to a first intermediate-result sub-block of an N-byte intermediate result;

(o) executing a logical exclusive-OR operation with a second seed sub-block and the first intermediate-result sub-block resulting from step (n) and sending a result of this logical exclusive-OR operation to a second intermediate-result sub-block of the N-byte intermediate result;

(p) executing modular addition with a j-th seed sub-block and a (j−1)-th intermediate-result sub-block and send-

ing a result of this modular addition to a j-th intermediate-result sub-block of the N-byte intermediate result;

(q) executing a logical exclusive-OR operation with a (j+1)-th seed sub-block and the j-th intermediate-result sub-block and sending a result of this logical exclusive-OR operation to a (j+1)-th intermediate-result sub-block of the N-byte intermediate result;

(r) carrying out steps (p) and (q) repeatedly for j=3, 5, 7, . . . , (N−1) until an N-th intermediate-result sub-block of the N-byte intermediate result is generated;

(s) executing a logical exclusive-OR operation with the N-byte intermediate result and an N-byte number having a random sequence of bits;

(t) executing on a result of step (s) a rotation operation to the left by L bits;

(u) assigning a result of step (t) to be a first round key having N-bytes;

(v) substituting the first round key for the previous N-byte seed key data for use as new N-byte seed key data and repeating steps (m) through (t) in order to generate a second round key; and

(w) repeating steps (m) through (t) to generate subsequent round keys, wherein each subsequent round key is generated by using the preceding round key as N-byte seed key data in step (m).

3. A computer useable medium having embodied thereon a computer program for executing a block cipher, the block cipher having a round process, the computer program being executable by a machine to perform the steps of:

(a) dividing a data stream into 2N-byte data blocks, each block being divided into a first half block and a second half block;

(b) executing a logical exclusive-OR operation with the second half block and an N-byte round key;

(c) dividing a result of step (b) into N divided blocks, sending a first divided block to a first S-box SI and sending to each remaining S-box S2, . . . , Sn a result of executing a logical exclusive-OR operation of each corresponding divided block with output data from the previous S-box;

(d) rotating an N-byte result of step (c) to the left by M bits;

(e) executing a logical exclusive-OR operation with the first half block and a result of step (d);

(f) relabeling the second half block as a new first half block for use in a next round, said next round utilizing a next round key;

(g) relabeling a result of step (e) as a new second half block for use in said next round;

(h) executing subsequent rounds by repeating steps (b) through (g) until just before a final round;

(i) sending a final second half block to a right half of a final output and executing a logical exclusive-OR operation with the final second half block and a final Nbyte round key;

(j) dividing a result of the logical exclusive-OR operation of step (i) into N final-round blocks, sending a first final-round block to the first S-box and sending to each remaining S-box a result of executing a logical exclusive-OR operation of each corresponding final-round block with output data from the previous S-box;

(k) rotating an N-byte result of step (j) to the left by M bits; and

(l) sending a result of executing a logical exclusive-OR operation with a final first half block and a result of step (k) to a left half of the final output.

4. The computer useable medium of claim 3, wherein said round keys are generated by a method comprising the steps of:

(m) breaking N-byte seed key data into N seed sub-blocks;

(n) executing modular addition with a first seed sub-block and an N-th seed sub-block and sending a result of this modular addition to a first intermediate-result sub-block of an N-byte intermediate result;

(o) executing a logical exclusive-OR operation with a second seed sub-block and the first intermediate-result sub-block resulting from step (n) and sending a result of this logical exclusive-OR operation to a second intermediate-result sub-block of the N-byte intermediate result;

(p) executing modular addition with a j-th seed sub-block and a (j−1)-th intermediate-result sub-block and sending a result of this modular addition to a j-th intermediate-result sub-block of the N-byte intermediate result;

(q) executing a logical exclusive-OR operation with a (j+1)-th seed sub-block and the j-th intermediate-result sub-block and sending a result of this logical exclusive-OR operation to a (j+1)-th intermediate-result sub-block of the N-byte intermediate result;

(r) carrying out steps (p) and (q) repeatedly for j=3, 5, 7, . . . , (N−1) until an N-th intermediate-result sub-block of the N-byte intermediate result is generated;

(s) executing a logical exclusive-OR operation with the N-byte intermediate result and an N-byte number having a random sequence of bits;

(t) executing on a result of step (s) a rotation operation to the left by L bits;

(u) assigning a result of step (t) to be a first round key having N-bytes;

(v) substituting the first round key for the previous N-byte seed key data for use as new N-byte seed key data and repeating steps (m) through (t) in order to generate a second round key; and

(w) repeating steps (m) through (t) to generate subsequent round keys, wherein each subsequent round key is generated by using the preceding round key as N-byte seed key data in step (m).

5. The block cipher method of claim 2, wherein said divided blocks, said finalround blocks, and said seed sub-blocks are 8 bits in length, wherein said modular addition is 256-modular addition, and wherein M=8 and L=5.

6. The block cipher method of claim 5, wherein said S-boxes S1, S2, . . . , Sn are selected from one of Type 1 or Type 2, wherein:

Type 1: S1=S2= . . . =Sn=f(x)

Type 2: S1=S3=S5= . . . =f(x)

S2=S4=S6= . . . =g(x),

wherein $f(x)=x^{-1}$ is an algebraic inversion of the Galois Field GF(256) and g(x)=h(h(x)) is the self-composition of the modular exponent function based 45, wherein $h(x)=45^x \bmod 257$.

7. The block cipher method of claim 1, wherein said divided blocks and said final-round blocks are 8 bits in length, and wherein M=8.

8. The block cipher method of claim 7, wherein said S-boxes S1, S2, . . . , Sn are selected from one of Type 1 or Type 2, wherein:

Type 1: S1=S2= . . . =Sn=f(x)

Type 2: S1=S3=S5= . . . =f(x)

S2=S4=S6= . . . =g(x),

wherein $f(x)=x^{-1}$ is an algebraic inversion of the Galois Field GF(256) and g(x)=h(h(x)) is the self-composition of the modular exponent function based 45, wherein $h(x)=45^x \bmod 257$.

9. The block cipher method of claim 1, wherein said round keys are generated by a method comprising the steps of:

(m) breaking a seed key into a plurality of N-byte seed key blocks;

(n) assigning a first N-byte seed key block to be a first N-byte round key;

(o) assigning a second N-byte seed key block to be a second N-byte round key;

(p) executing a function using the second N-byte seed key block as input into the function, the function producing N-byte output;

(q) executing a logical exclusive-OR operation with a result of step (p) and a fourth N-byte seed key block;

(r) assigning a result of step (q) to be a third N-byte round key;

(s) executing the function using the first N-byte seed key block as input into the function;

(t) executing a logical exclusive-OR operation with a result of step (s) and a third N-byte seed key block;

(u) assigning a result of step (t) to be a fourth round key; and

(v) generating remaining round keys, wherein an (i+1)-th round key is generated by executing a logical exclusive-OR operation with an (i−1)-th round key and a result of executing the function on an i-th round key, wherein the remaining round keys are generated for i=4, 5, 6, etc. until a final round key is generated.

10. The block cipher method of claim 9, wherein executing the function comprises the steps of:

(1) breaking N-byte seed-key data into N seed sub-blocks;

(2) executing modular-addition with a first seed sub-block and an N-th seed sub-block and sending a result of this modular addition to a first intermediate-result sub-block of an N-byte intermediate result;

(3) executing a logical exclusive-OR operation with a second seed sub-block and the first intermediate-result sub-block resulting from step (2) and sending a result of this logical exclusive-OR operation to a second intermediate-result sub-block of the N-byte intermediate result;

(4) executing modular addition with a j-th seed sub-block and a (j−1)-th intermediate-result sub-block and sending a result of this modular addition to a j-th intermediate-result sub-block of the N-byte intermediate result;

(5) executing a logical exclusive-OR operation with a (j+1)-th seed sub-block and the j-th intermediate-result sub-block and sending a result of this logical exclusive-OR operation to a (j+1)-th intermediate-result sub-block of the N-byte intermediate result;

(6) carrying out steps (4) and (5) repeatedly for j=3, 5, 7, . . . , (N−1) until an N-th intermediate-result sub-block is generated;

(7) executing a logical exclusive-OR operation with the N-byte intermediate result and an N-byte number having a random sequence of bits; and

(8) executing on a result of step (7) a rotation operation to the left by L bits.

11. The block cipher method of claim 10, wherein said divided blocks, said final-round blocks, and said seed sub-blocks are 8 bits in length, wherein said modular addition is 256-modular addition, and wherein M=8 and L=5.

12. The computer useable medium of claim 4, wherein said divided blocks, said final-round blocks, and said seed sub-blocks are 8 bits in length, wherein said modular addition is 256-modular addition, and wherein M=8 and L=5.

13. The computer useable medium of claim 3, wherein said divided blocks and said final-round blocks are 8 bits in length, and wherein M=8.

14. The computer useable medium of claim 13, wherein the round process repeats sixteen times and wherein sixteen round keys are utilized.

15. The computer useable medium of claim 4, wherein said N-byte number having a random sequence of bits is "0xb7e15163" and wherein N=4.

16. A block cipher method, comprising the steps of:

(a) dividing a data stream into blocks, each block being divided into a first half block and a second half block;

(b) combining the second half block with a round key using a first logical operation;

(c) dividing a result of step (b) into sub-blocks, using a first sub-block to provide input into a first S-box, and using remaining sub-blocks to provide input into remaining S-boxes, wherein a remaining sub-block is combined with output from a remaining S-box using a second logical operation, and wherein the result of the second logical operation is provided as input into another S-box;

(d) permuting the outputs from the S-boxes resulting from step (c);

(e) combining a result of step (d) with the first half block using a third logical operation;

(f) relabeling the second half block as a new first half block for use in a next round, the next round utilizing a next round key;

(g) relabeling a result of step (e) as a new second half block for use in the next round;

(h) repeating steps (b) through (g) for subsequent rounds;

(i) generating a right half of a final output using a final second half block; and

(j) generating a left half of the final output using a final first half block.

17. The block cipher method of claim 16, wherein the first, second and third logical operations are selected from the group consisting of exclusive-OR and modular addition.

18. The block cipher method of claim 17, wherein permuting the outputs of the S-boxes referred to in step (d) comprises rotating the outputs of the S-boxes by a predetermined number of bits.

19. The block cipher method of claim 18, wherein the second logical operation is the exclusive-OR operation, and wherein the input to the k-th S-box in step (c) is given by $X(k) \oplus S(k-1)$, wherein

X(k) represents the k-th sub-block,

S(k−1) represents the output from (k−1)-th S-box,

$\oplus$ represents the exclusive-OR operation, and

k>1.

20. The block cipher method of claim 19, wherein the first and third logical operations are exclusive-OR operations.

21. A method of generating round keys, comprising the steps of:

(a) breaking a seed key into a plurality of seed key blocks;

(b) assigning a first seed key block to be a first round key, K(1);

(c) assigning a second seed key block to be a second round key, K(2);

(d) generating subsequent round keys using output from a key-generating function (FUNCTION) wherein at least some of said subsequent round keys are generated according to the relation K(i+1)=FUNCTION(K(i))□K(i−1), wherein

K(i+1) is the (i+1)-th round key,

K(i−1) is the (i−1)-th round key,

K(i) is the i-th round key,

i≠1, and

□ represents a first logical operation that combines FUNCTION(K(i)) and K(i−1).

22. The method of claim 21, wherein the first logical operation □ is one of an exclusive-OR operation and modular addition.

23. The method of claim 22, wherein the key-generating function (FUNCTION) performs steps including:

(e) breaking input data into sub-blocks, Q(j);

(f) combining sub-blocks using second and third logical operations to generate intermediate sub-blocks, T(j);

(g) combining the result of step (f) with a number having a random sequence of bits using a fourth logical operation; and

(h) permuting the result of step (g).

24. The method of claim 23, wherein the combining in step (f) generates intermediate sub-blocks, T(j), according to the relation T(j)=Q(j)◇T(j−1), wherein

◇ represents either of the second and third logical operations, and wherein j>1.

25. The method of claim 24, wherein the second, third, and fourth logical operations are selected from the group consisting of exclusive-OR and modular addition.

26. The method of claim 25, wherein said permuting in step (h) comprises rotating by a predetermined number of bits.

* * * * *

28/3,K/32 (Item 32 from file: 350)
DIALOG(R)File 350:Derwent WPIX

013889191 **Image available**
WPI Acc No: 2001-373404/200139
XRPX Acc No: N01-273095
Carrying out symmetric key block cipher using multiple stages has operands of arithmetic operations are word being used to encipher **selected data block and generated key value having length identical to the word**
Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: COPPERSMITH D; GENNARO R; HALEVI S; JUTLA C S; MATYAS S M; O'CONNOR L J; PEYRAVIAN M; SAFFORD D R; ZUNIC N
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 6185679	B1	20010206	US 9827769	A	19980223	200139	B

Priority Applications (No Type Date): US 9827769 A 19980223
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6185679	B1		24	H04L-009/00	

... **block cipher using multiple stages has operands of arithmetic operations are word being used to** encipher **selected data block and generated key value having length identical to the word**

Abstract (Basic):
... A **first** simple arithmetic operation is performed in a **first** stage; a Type-3 Feistel unkeyed mixing operation in **second** stage; a Type-1 Feistel keyed data-dependent rotation transformed in third stage; Type-3 Feistel unkeyed inverse mixing operation in fourth stage; and a **second** simple arithmetic operation is performed which is identical to the **first** arithmetic operation in fifth stage.
... The **first** and **second** arithmetic operations are identical and are one of (1) an addition operation, (2) a subtraction...

...exclusive OR operation. The operands of the arithmetic operations are a word being used to **encipher** a selected data block and a generated key value which has a length identical to...

...For a symmetric key block cipher, which uses multiple stages, where the stages have **different** structures and **different** subround functions ...

...lookup, exclusive OR, addition, subtraction, and data-dependent rotation, which minimizes the time required to **encrypt** and **decrypt** data. Data dependent rotation is fast because of the manner in which the data value in a single predetermined register is located. Table lookup using **S - boxes** is faster because some rounds access the **S - boxes** without using subkeys, which are data dependent and can be precomputed further minimizing the time required for **encryption** and **decryption** and a minimal amount of computer storage is required for data used in the operation...

...The figure shows an illustration of the stages of operation used for **encrypting** a block of plaintext into a block of ciphertext...
...Title Terms: **ENCIPHER** ;
International Patent Class (Main): **H04L-009/00**
Manual Codes (EPI/S-X): **T01-D01** ...

... T01-E02A ...

... T01-E02C ...

... T01-E02D ...

... T01-J12C ...

... T01-S01B ...

... T01-S03 ...

... W01-A05A

US006185679B1

(12) **United States Patent**
Coppersmith et al.

(10) **Patent No.: US 6,185,679 B1**
(45) **Date of Patent: Feb. 6, 2001**

(54) **METHOD AND APPARATUS FOR A SYMMETRIC BLOCK CIPHER USING MULTIPLE STAGES WITH TYPE-1 AND TYPE-3 FEISTEL NETWORKS**

(75) Inventors: **Don Coppersmith**, Ossining; **Rosario Gennaro**, New York; **Shai Halevi**, Heartsdale; **Charanjit S. Jutla**, Elmsford, all of NY (US); **Stephen M. Matyas, Jr.**, Manassas, VA (US); **Luke James O'Connor**, Adliswil (CH); **Mohammed Peyravian**, Cary, NC (US); **David Robert Safford**, Brewster; **Nevenko Zunic**, Wappingers Falls, both of NY (US)

(73) Assignee: **International Business Machines Corporation**, Armonk, NY (US)

(*) Notice: Under 35 U.S.C. 154(b), the term of this patent shall be extended for 0 days.

(21) Appl. No.: **09/027,769**

(22) Filed: **Feb. 23, 1998**

(51) **Int. Cl.**[7] **H04L 9/00**
(52) **U.S. Cl.** **713/150**; 380/37
(58) **Field of Search** 380/28, 29, 42, 380/37

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,157,454 * 6/1979 Becker 178/22
5,724,428 * 3/1998 Rivest 380/37

OTHER PUBLICATIONS

B. Schneier, Description of a New Variable–Length Key, 64–Bit Block Cipher (Blowfish), Dec. 1993.*
Y. Zhen, On the Construction of Block Ciphers Provably Secure and Not Relying on Any Uproved Hypothesis, 1989.*
B. Schneier, et al. Unbalanced Feistel Networks and Block–Cipher Design, 1996.*
B. Schneier, Applied Cryptography 2e John Wiley pp. 193, 198–201, 347, 1996.*

* cited by examiner

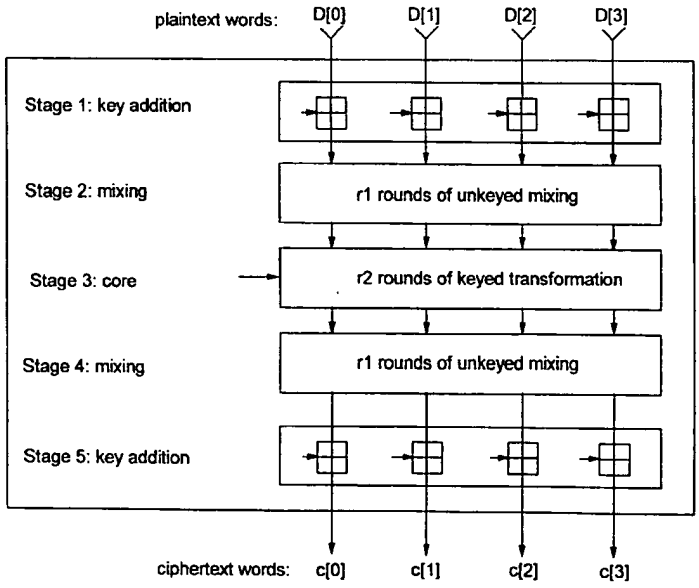
Primary Examiner—Gail O. Hayes
Assistant Examiner—James Seal
(74) *Attorney, Agent, or Firm*—Jeanine S. Ray-Yarletts; Marcia L. Doubet

(57) **ABSTRACT**

The present invention provides a technique, system, and computer program for a symmetric key block cipher. Variable block sizes and key sizes are supported, as well as a variable number of rounds. The cipher uses multiple stages of processing, where the stages have different structures and different subround functions, to provide excellent resistance to both linear and differential attacks. Feistel Type-1 and Type-3 are both used, each during different stages. The number of rounds may vary among stages. Subkeys are used in some, but not all, stages. The variable-length keys can be precomputed. A novel manner of using data-dependent rotation in a cipher is defined.

25 Claims, 9 Drawing Sheets

The general structure of the encryption operation



-continued

```
D[0] = IROTATE (D[0], D[1], w);
D[1] -= D[3];
D[0] ^= D[1];
D[1] -= E[n--];                    /* begin subround 4*/
D[1] = IROTATE (D[1], D[7], w);
D[1] ^= D[2];
}
```

As will be understood by one skilled in the art, this "C" language code specifies the processing for 2 rounds. The first set of statements is used for an even-numbered round, and the second set of statements is used for an odd-numbered round. As previously indicated, these statements correspond to the diagrams shown for encryption in FIGS. 6B and 6A, respectively, if those diagrams are read from the bottom up, addition is changed to subtraction, and left rotation is changed to right rotation.

The stage begins by first initializing the subkey index, to point to the last subkey from a group of 8 subkeys that were used during encryption for one iteration through the Stage 3 processing. Then processing for an even-numbered round of decryption begins, and that subkey value is subtracted from D[3]. The subkey index is decremented. Then, D[3] is rotated to the right, where the amount of rotation is determined using the value in D[1]. Next, the value of D[0] is subtracted from D[1]. Finally, this subround performs an exclusive OR operation, where the two operands are the data words D[2] and D[3]. The result becomes the new value of D[3].

In the second subround, the next-preceding subkey (that is, the one indexed by the previously-decremented value) is subtracted from D[2], and the index is again decremented The new value of D[2] is then rotated to the right, with the amount of rotation again specified by the value in D[1]. The value in D[3] is then subtracted from D[1]. Finally, D[0] is exclusive OR'd with D[2], forming a new value for D[2].

The third and fourth subrounds for the even-numbered rounds are similar. In the third subround, the next preceding subkey is subtracted from D[0]; D[0] is rotated to the right by the amount indicated by D[1]; D[2] is subtracted from D[1]; and D[1] is exclusive OR'd with D[0], forming a new D[0]. In the fourth subround, the next preceding subkey is subtracted from D[1]; D[1] is rotated 7 positions to the right; and D[3] is exclusive OR'd with D[1], forming a new value for D[1].

By the end of this even-numbered round, all of the data words have been rotated, the value of each data word has impacted a rotation operation, each data word has been exclusive OR'd with another data word, and each data word has a new value.

For an odd-numbered round, the processing is similar to that just described for even rounds. The only difference is the order in which the different data words are used by the operations, as shown by the "C" language statements.

While the preferred embodiment of the present invention has been described, additional variations and modifications in that embodiment may occur to those skilled in the art once they learn of the basic inventive concepts. Therefore, it is intended that the appended claims shall be construed to include both the preferred embodiment and all such variations and modifications as fall within the spirit and scope of the invention.

We claim:

1. A method of carrying out a symmetric key block cipher using multiple stages, comprising the steps of:

performing a first simple arithmetic operation in a first stage;

performing a Type-3 Feistel unkeyed mixing operation in a second stage;

performing a Type-1 Feistel keyed data-dependent rotation transform in a third stage, wherein a fixed location is used to specify an amount of the data-dependent rotation for each of a plurality of rounds of the third stage;

performing a Type-3 Feistel unkeyed inverse mixing operation in a fourth stage; and

performing a second simple arithmetic operation in a fifth stage, wherein the first simple arithmetic operation and the second simple arithmetic operation may be identical.

2. The method according to claim 1, wherein one or more of the steps is embodied in a hardware chip.

3. The method according to claim 1, wherein:

the first simple arithmetic operation is one of (1) an addition operation, (2) a subtraction operation, or (3) an exclusive OR operation; and

the second simple arithmetic operation is one of (1) the addition operation, (2) the subtraction operation, or (3) the exclusive OR operation.

4. The method according to claim 3, wherein operands of the first and second simple arithmetic operations are a word being used to encipher a selected data block and a generated key value which has a length identical to that of the word.

5. The method according to claim 1, wherein a plurality of first feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed mixing operation, and a plurality of second feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed inverse mixing operation.

6. The method according to claim 5, wherein the first feedback operation is an addition operation and the second feedback operation is a subtraction operation.

7. The method according to claim 1, wherein the Type-3 Feistel unkeyed mixing operation and the Type-3 Feistel unkeyed inverse mixing operation retrieve values from 2 substitution boxes.

8. The method according to claim 1, wherein a round function used in each of a plurality of subrounds of the plurality of rounds of the third stage for the Type-1 Feistel keyed data-dependent rotation transform comprises the steps of:

performing an exclusive OR operation using two selected ones of a plurality of words from a block being enciphered, wherein one of the two selected ones is a word being transformed by a current one of the subrounds;

adding, except in an initial even-numbered one of the plurality of subrounds and an initial odd-numbered one of the plurality of subrounds, a round-specific one of the plurality of words to a predetermined one of the plurality, wherein the predetermined one stays constant throughout all of the plurality of subrounds for all of the plurality of rounds; and

performing the data-dependent rotation operation on the word being transformed, using the predetermined one as the fixed location to specify the amount of the data-dependent rotation operation, except in the initial even-numbered and initial odd-numbered subrounds which use a fixed value for the amount.

9. The method according to claim 1, wherein the cipher supports a variable number of rounds in at least one of the

stages, a variable length of generated key values to be used with the cipher in at least the Type-1 Feistel keyed data-dependent rotation transform, and a variable length of input blocks to be enciphered.

10. A system for carrying out a symmetric key block cipher using multiple stages on a computer, comprising:

means for performing a first simple arithmetic operation in a first stage;

means for performing a Type-3 Feistel unkeyed mixing operation in a second stage;

means for performing a Type-1 Feistel keyed data-dependent rotation transform in a third stage, wherein a fixed location is used to specify an amount of the data-dependent rotation for each of a plurality of rounds of the third stage,

means for performing a Type-3 Feistel unkeyed inverse mixing operation in a fourth stage; and

means for performing a second simple arithmetic operation in a fifth stage, wherein the first simple arithmetic operation and the second simple arithmetic operation may be identical.

11. The system according to claim **10**, wherein:

the first simple arithmetic operation is one of (1) an addition operation, (2) a subtraction operation, or (3) an exclusive OR operation; and

the second simple arithmetic operation is one of (1) the addition operation, (2) the subtraction operation, or (3) the exclusive OR operation.

12. The system according to claim **11**, wherein operands of the first and second simple arithmetic operations are a word being used to encipher a selected data block and a generated key value which has a length identical to that of the word.

13. The system according to claim **10**, wherein a plurality of first feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed mixing operation, and a plurality of second feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed inverse mixing operation.

14. The system according to claim **13**, wherein the first feedback operation is an addition operation and the second feedback operation is a subtraction operation.

15. The system according to claim **10**, wherein the Type-3 Feistel unkeyed mixing operation and the Type-3 Feistel unkeyed inverse mixing operation retrieve values from 2 substitution boxes.

16. The system according to claim **10**, wherein a round function used in each of a plurality of subrounds of the plurality of rounds of the third stage for the Type-1 Feistel keyed data-dependent rotation transform comprises:

means for performing an exclusive OR operation using two selected ones of a plurality of words from a block being enciphered, wherein one of the two selected ones is a word being transformed by a current one of the subrounds;

means for adding, except in an initial even-numbered one of the plurality of subrounds and an initial odd-numbered one of the plurality of subrounds, a round-specific one of the plurality of words to a predetermined one of the plurality, wherein the predetermined one stays constant throughout all of the plurality of subrounds for all of the plurality of rounds; and

means for performing the data-dependent rotation operation on the word being transformed, using the predetermined one as the fixed location to specify the amount of the data-dependent rotation operation, except in the initial even-numbered and initial odd-numbered subrounds which use a fixed value for the amount.

17. The system according to claim **10**, wherein the cipher supports a variable number of rounds in at least one of the stages, a variable length of generated key values to be used with the cipher in at least the Type-1 Feistel keyed data-dependent rotation transform, and a variable length of input blocks to be enciphered.

18. A computer program product for carrying out a symmetric key block cipher using multiple stages on a computer, the computer program product embodied in a computer-readable medium and comprising:

computer-readable program code means for performing a first simple arithmetic operation in a first stage;

computer-readable program code means for performing a Type-3 Feistel unkeyed mixing operation in a second stage;

computer-readable program code means for performing a Type-1 Feistel keyed data-dependent rotation transform in a third stage, wherein a fixed location is used to specify an amount of the data-dependent rotation for each of a plurality of rounds of the third stage;

computer-readable program code means for performing a Type-3 Feistel unkeyed inverse mixing operation in a fourth stage; and

computer-readable program code means for performing a second simple arithmetic operation in a fifth stage, wherein the first simple arithmetic operation and the second simple arithmetic operation may be identical.

19. The computer program product according to claim **18**, wherein:

the first simple arithmetic operation is one of (1) an addition operation, (2) a subtraction operation, or (3) an exclusive OR operation; and

the second simple arithmetic operation is one of (1) the addition operation, (2) the subtraction operation, or (3) the exclusive OR operation.

20. The computer program product according to claim **19**, wherein operands of the first and second simple arithmetic operations are a word being used to encipher a selected data block and a generated key value which has a length identical to that of the word.

21. The computer program product according to claim **18**, wherein a plurality of first feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed mixing operation, and a plurality of second feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed inverse mixing operation.

22. The computer program product according to claim **21**, wherein the first feedback operation is an addition operation and the second feedback operation is a subtraction operation.

23. The computer program product according to claim **18**, wherein the Type-3 Feistel unkeyed mixing operation and the Type-3 Feistel unkeyed inverse mixing operation retrieve values from 2 substitution boxes.

24. The computer program product according to claim **18**, wherein a round function used in each of a plurality of subrounds of the plurality of rounds of the third stage for the Type-1 Feistel keyed data-dependent rotation transform comprises:

computer-readable program code means for performing an exclusive OR operation using two selected ones of

a plurality of words from a block being enciphered, wherein one of the two selected ones is a word being transformed by a current one of the subrounds;

computer-readable program code means for adding, except in an initial even-numbered one of the plurality of subrounds and an initial odd-numbered one of the plurality of subrounds, a round-specific one of the plurality of words to a predetermined one of the plurality, wherein the predetermined one stays constant throughout all of the plurality of subrounds for all of the plurality of rounds; and

computer-readable program code means for performing the data-dependent rotation operation on the word being transformed, using the predetermined one as the fixed location to specify the amount of the data-dependent rotation operation, except in the initial even-numbered and initial odd-numbered subrounds which use a fixed value for the amount.

25. The computer program product according to claim **18**, wherein the cipher supports a variable number of rounds in at least one of the stages, a variable length of generated key values to be used with the cipher in at least the Type-1 Feistel keyed data-dependent rotation transform, and a variable length of input blocks to be enciphered.

* * * * *

28/3,K/30 (Item 30 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014028614 **Image available**
WPI Acc No: 2001-512828/200156
XRPX Acc No: N01-379684
Encrypting **input file containing number of blocks using symmetric key block cipher having odd number of stages has simple arithmetic operation performed in** first **even number of stages**
Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: COPPERSMITH D; GENNARO R; HALEVI S; JUTLA C S; MATYAS S M; O'CONNOR L J; PEYRAVIAN M; SAFFORD D R; ZUNIC N
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 6185304	B1	20010206	US 9827765	A	19980223	200156	B

Priority Applications (No Type Date): US 9827765 A 19980223
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6185304	B1		23	H04K-001/00	

Encrypting **input file containing number of blocks using symmetric key block cipher having odd number of stages has simple arithmetic operation performed in** first **even number of stages**

Abstract (Basic):
... A simple arithmetic operation is performed in a **first** even number of stages; in a **second** even number of stages performing an identical number of: (1) Type-3 Feistel unkeyed mixing...

...4) and (2) Type-3 Feistel unkeyed inverse mixing operation which retrieve values from 2 **substitution boxes** . The **first** and **second** even numbers can be identical; and a Type-3 Feistel keyed transform in a remaining...
... For a symmetric key block cipher, which uses multiple stages, where the stages have **different** structures and **different** subround functions. The cipher allows the block size, key size, and number of rounds per...

...Provides a flexible symmetric block cipher which offers excellent resistance to linear and **differential** attacks; operates quickly and efficiently while using **S - boxes** ; uses multiplication in a fast and efficient round function because of using an algebraic ring...

...per stage. The data-independent subkeys can be precomputed, further minimizing the time required for **encryption** and **decryption** and a minimal amount of computer storage is required for data used in the operation...

...The figure shows the stages of operation used for **encrypting** a block of plain text into a block of cipher text...
...Title Terms: **FIRST** ;
International Patent Class (Main): **H04K-001/00**
Manual Codes (EPI/S-X): **T01-D01** ...

... **T01-E02D** ...

... **T01-S01B** ...

... T01-S03 ...

... W01-A05A

US006185304B1

(12) **United States Patent**
Coppersmith et al.

(10) **Patent No.: US 6,185,304 B1**
(45) **Date of Patent: Feb. 6, 2001**

(54) **METHOD AND APPARATUS FOR A SYMMETRIC BLOCK CIPHER USING MULTIPLE STAGES**

(75) Inventors: **Don Coppersmith**, Ossining; **Rosario Gennaro**, New York; **Shai Halevi**, Heartsdale; **Charanjit S. Jutla**, Elmsford, all of NY (US); **Stephen M. Matyas, Jr.**, Manassas, VA (US); **Luke James O'Connor**, Adliswil (CH); **Mohammed Peyravian**, Cary, NC (US); **David Robert Safford**, Brewster; **Nevenko Zunic**, Wappingers Falls, both of NY (US)

(73) Assignee: **International Business Machines Corporation**, Armonk, NY (US)

(*) Notice: Under 35 U.S.C. 154(b), the term of this patent shall be extended for 0 days.

(21) Appl. No.: **09/027,765**

(22) Filed: **Feb. 23, 1998**

(51) **Int. Cl.**[7] **H04K 1/00**
(52) **U.S. Cl.** **380/37**; 380/259
(58) **Field of Search** 380/28, 29, 37, 380/42

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,157,454	*	6/1979	Becker	178/22
5,511,123	*	4/1996	Adams	380/29
5,724,428	*	3/1998	Rivest	380/28

OTHER PUBLICATIONS

Even and Goldreich, On The Power of Cascade Ciphers, 1983.*

John Savard, Cypto Compendium, httpo://home ecn.ab.ca/~jsavard/crypto/co041203.htm, 1998.*

Matthew Kwan, The Design of the ICE encrytpion Algorithm, Proceeding of Fast Software, 1997.*

B. Schneier, Description of a New Variable–Length Key, 64–Bit Block Cipher (Bowfish), Dec. 1993.*

Y. Zheng, On the Construction of Block Ciphers Provably Secure and Not Relying on Any Unproved Hypotheses, 1989.*

B. Schneier, et. al., Unbalanced Feistel Networks an Block–Cipher Design, 1996.*

B. Schneier, Applied Cryptography, 2e, John Wiley pp. 193, 198–201, 347, 1995.*

Adina Di Porto, Vino: A Block Cipher including Variable Permutations, 1993.*

* cited by examiner

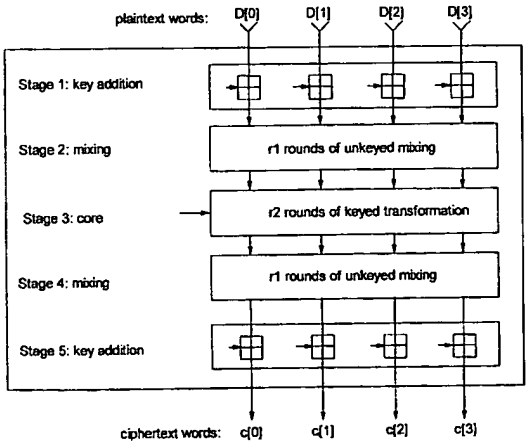
Primary Examiner—Gail O. Hayes
Assistant Examiner—James Seal
(74) *Attorney, Agent, or Firm*—Jeanine S. Ray-Yarletts; Marcia L. Doubet

(57) **ABSTRACT**

The present invention provides a technique, system, and computer program for a symmetric key block cipher. Variable block sizes and key sizes are supported, as well as a variable number of rounds. The cipher uses multiple stages of processing, where the stages have different structures and different subround functions, to provide excellent resistance to both linear and differential attacks. Feistel Type-3 networks are used, with different networks during different stages. The number of rounds may vary among stages. Subkeys are used in some, but not all, stages. The variable-length keys can be precomputed. A novel manner of using multiplication in a cipher is defined.

29 Claims, 9 Drawing Sheets

The general structure of the encryption operation



stored in temp1, temp2, and temp3, are used to modify the other 3 data words. These three statements are inverted from their corresponding encryption statement. Note that since the statements are independent of one another, the order of the three statements has not been inverted herein. Alternatively, the order could be inverted, without changing the functioning of the statements. The data word passed as the first of these 3 input parameters, D[out1], is modified by having the value in temp1 subtracted from it. The data word passed as the second of these 3 parameters, D[out2], is modified by being exclusive OR'd with the value in temp2. The data word passed as the third of these 3 parameters, D[out3], is modified by having the value in temp3 subtracted from it. The processing of the inverseSubRound function is now complete for this subround. All other subrounds are processed in an identical manner.

While the preferred embodiment of the present invention has been described, additional variations and modifications in that embodiment may occur to those skilled in the art once they learn of the basic inventive concepts. Therefore, it is intended that the appended claims shall be construed to include both the preferred embodiment and all such variations and modifications as fall within the spirit and scope of the invention.

We claim:

1. A method of encrypting an input file comprising a plurality of blocks using a symmetric key block cipher having an odd number of stages, wherein the odd number is at least 5, comprising the steps of:

performing a simple arithmetic operation in a first even number of the stages;

performing, in a second even number of the stages, an identical number of: (1) a Type-3 Feistel unkeyed mixing operation and (2) a Type-3 Feistel unkeyed inverse mixing operation, wherein the first even number and the second even number may be identical; and

performing a Type-3 Feistel keyed transform in a remaining number of the stages.

2. The method according to claim 1, wherein the stages in a first half of the first even number and of the second even number are performed during a first half of the odd number of stages, and the stages in a second half of the first even number and of the second even number are performed during a second half of the odd number of stages.

3. A method of carrying out a symmetric key block cipher using multiple stages, comprising the steps of:

performing a first simple arithmetic operation in a first stage;

performing a Type-3 Feistel unkeyed mixing operation in a second stage;

performing a Type-3 Feistel keyed transform in a third stage;

performing a Type-3 Feistel unkeyed inverse mixing operation in a fourth stage; and

performing a second simple arithmetic operation in a fifth stage, wherein the first simple arithmetic operation and the second simple arithmetic operation may be identical.

4. The method according to claim 3, wherein one or more of the steps is embodied in a hardware chip.

5. The method according to claim 3, wherein:

the first simple arithmetic operation is one of (1) an addition operation, (2) a subtraction operation, or (3) an exclusive OR operation; and

the second simple arithmetic operation is one of (1) the addition operation, (2) the subtraction operation, or (3) the exclusive OR operation.

6. The method according to claim 5, wherein operands of the first and second simple arithmetic operations are a word being used to encipher a selected data block and a generated key value which has a length identical to that of the word.

7. The method according to claim 3, wherein a plurality of first feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed mixing operation, and a plurality of second feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed inverse mixing operation.

8. The method according to claim 7, wherein the first feedback operation is an addition operation and the second feedback operation is a subtraction operation.

9. The method according to claim 3, wherein the Type-3 Feistel unkeyed mixing operation and the Type-3 Feistel unkeyed inverse mixing operation retrieve values from 2 substitution boxes.

10. The method according to claim 3, wherein a round function of the Type-3 Feistel keyed transform comprises a forward function using (1) an integer multiplication modulo 2^x operation with a generated key value, where x is a bit length of a word from a block being enciphered, and (2) a data-dependent rotation operation.

11. The method according to claim 3, wherein the cipher supports a variable number of rounds in at least one of the stages, a variable length of generated key values to be used with the cipher in at least the Type-3 Feistel keyed transform, and a variable length of input blocks to be enciphered.

12. A system for carrying out a symmetric key block cipher using multiple stages, comprising:

means for performing a simple arithmetic operation in a first stage;

means for performing a Type-3 Feistel unkeyed mixing operation in a second stage;

means for performing a Type-3 Feistel keyed transform in a third stage;

means for performing a Type-3 Feistel unkeyed inverse mixing operation in a fourth stage; and

means for performing the simple arithmetic operation in a fifth stage.

13. The system according to claim 12, wherein the simple arithmetic operation is one of (1) an addition operation, (2) a subtraction operation, or (3) an exclusive OR operation.

14. The system according to claim 13, wherein operands of the simple arithmetic operation are a word being used to encipher a selected data block and a generated key value which has a length identical to that of the word.

15. The system according to claim 12, wherein a plurality of first feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed mixing operation, and a plurality of second feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed inverse mixing operation.

16. The system according to claim 15, wherein the first feedback operation is an addition operation and the second feedback operation is a subtraction operation.

17. The system according to claim 12, wherein the Type-3 Feistel unkeyed mixing operation and the Type-3 Feistel unkeyed inverse mixing operation retrieve values from 2 substitution boxes.

18. The system according to claim 12, wherein a round function of the Type-3 Feistel keyed transform comprises a forward function using (1) an integer multiplication modulo 2^x operation with a generated key value, where x is a bit length of a word from a block being enciphered, and (2) a data-dependent rotation operation.

19. The system according to claim 12, wherein the cipher supports a variable number of rounds in at least one of the stages, a variable length of generated key values to be used with the cipher in at least the Type-3 Feistel keyed transform, and a variable length of input blocks to be enciphered.

20. The system according to claim 12, wherein one or more of the means is embodied in a hardware chip.

21. A computer program product for carrying out a symmetric key block cipher using multiple stages with a computer, the computer program product embodied on a computer-readable medium and comprising:

computer-readable program code means for performing a simple arithmetic operation in a first stage;

computer-readable program code means for performing a Type-3 Feistel unkeyed mixing operation in a second stage;

computer-readable program code means for performing a Type-3 Feistel keyed transform in a third stage,

computer-readable program code means for performing a Type-3 Feistel unkeyed inverse mixing operation in a fourth stage; and

computer-readable program code means for performing the simple arithmetic operation in a fifth stage.

22. The computer program product according to claim 21, wherein the simple arithmetic operation is one of (1) an addition operation, (2) a subtraction operation, or (3) an exclusive OR operation.

23. The computer program product according to claim 22, wherein operands of the simple arithmetic operation are a word being used to encipher a selected data block and a generated key value which has a length identical to that of the word.

24. The computer program product according to claim 21, wherein a plurality of first feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed mixing operation, and a plurality of second feedback operations are performed between distinct rounds of the Type-3 Feistel unkeyed inverse mixing operation.

25. The computer program product according to claim 24, wherein the first feedback operation is an addition operation and the second feedback operation is a subtraction operation.

26. The computer program product according to claim 21, wherein the Type-3 Feistel unkeyed mixing operation and the Type-3 Feistel unkeyed inverse mixing operation retrieve values from 2 substitution boxes.

27. The computer program product according to claim 21, wherein a round function of the Type-3 Feistel keyed transform comprises a forward function using (1) an integer multiplication modulo 2^x operation with a generated key value, where x is a bit length of a word from a block being enciphered, and (2) a data-dependent rotation operation.

28. The computer program product according to claim 21, wherein the cipher supports a variable number of rounds in at least one of the stages, a variable length of generated key values to be used with the cipher in at least the Type-3 Feistel keyed transform, and a variable length of input blocks to be enciphered.

29. The computer program product according to claim 21, wherein one or more of the computer-readable program code means is embodied in a hardware chip.

* * * * *

```
Set     Items   Description
S1      51158   (DIGITAL? OR MULTIMEDIA?)()(GOOD? OR PRODUCT? OR ENTIT? OR
             MODULE? OR UNIT? OR DEVICE?) OR MPEG? ? OR MP3? ?
S2    2132595   SOFTWARE? OR NONSOFTWARE? OR DVD? ? OR CDROM? OR CD()ROM? ?
              OR DISK? OR DISC? ? OR FLOPPY? OR FLOPPIE?
S3       3906   (AUDIO?(10N)VIDEO?)(2N)(DATA? OR GOOD? OR PRODUCT? OR MODU-
             LE? OR ENTIT? OR UNIT? OR DEVICE?)
S4       1919   SBOX? OR S()(BOX OR BOXES)
S5          4   RIJNDAELSBOX? OR S()FUNCTION?()(BOX OR BOXES)
S6          0   (SOFTWARE? OR SOFT()WARE?)()USAGE?()MONITOR?()(BOX OR BOXE-
             S) OR SUBSTITUT?()FUNCTION?()(BOX OR BOXES)
S7        149   SUBSTITUTION?()(LOOKUP OR LOOK?()UP)()TABLE? OR SUBSTITUT?-
             ()LUT? ? OR SUBSTITUT?()(BOX OR BOXES)
S8          0   RANDOM?()SUBSTITUT?()(LUT? ? OR BOX OR BOXES OR TABLE?)
S9    5605088   PARTITION? OR PART? ? OR PARTIAL? OR SEGMENT? OR DIVISION?
S10   2079963   PARCEL? OR PIECE? OR CHUNK? OR FRACTION? OR SLICE? OR DIVI-
             D?
S11   2321505   SECTION? OR SECTOR? OR PORTION? OR APPORTION? OR SECTOR?
S12   9333243   FIRST? OR 1ST OR PRIMARY OR INITIAL? OR ORIGINAL? OR LEADO-
             FF? OR MAIN OR CHIEF OR INTRODUCTORY? OR MASTER?
S13    822246   SUBSTITUT? OR PROXIE? OR PROXY? OR STANDIN OR STANDINS OR -
             STAND?()IN
S14  14541498   SECOND? OR 2ND OR DOUBL? OR TWIN? OR EXTRA? OR ANOTHER OR -
             SUBSIDIAR? OR AUXILIAR? OR DIFFERENT? OR ALTERNAT? OR SLAVE?
S15    244213   ENCRYPT? OR ENCIPHER? OR ENCYPHER? OR SCRAMBL? OR HASH? OR
             CRYPT? OR ENSCRAMBL?
S16     15300   DECRYPT? OR DECIPHER? OR DECYPHER? OR DESCRAMBL? OR DEHASH?
              OR UNSCRAMBL? OR UNENCRYPT?
S17         0   IC=(H04K? OR H04L?)
S18         0   MC=(T01? OR W01? OR W04?)
S19        13   S1:S3 AND S4:S8 AND S9:S11 AND S15:S16
S20         2   S19 AND S12:S14(7N)(S1:S3 OR S9:S11)
S21        13   S19:S20
S22         6   S21 AND PY<2001
S23         6   RD  (unique items)
File   2:INSPEC 1898-2006/Dec W4
         (c) 2006 Institution of Electrical Engineers
File   6:NTIS 1964-2006/Jan W2
         (c) 2006 NTIS, Intl Cpyrght All Rights Res
File   8:Ei Compendex(R) 1970-2006/Jan W2
         (c) 2006 Elsevier Eng.  Info. Inc.
File  34:SciSearch(R) Cited Ref Sci 1990-2006/Jan W2
         (c) 2006 Inst for Sci Info
File  35:Dissertation Abs Online 1861-2005/Dec
         (c) 2005 ProQuest Info&Learning
File  65:Inside Conferences 1993-2006/Jan W3
         (c) 2006 BLDSC all rts. reserv.
File  94:JICST-EPlus 1985-2006/Nov W1
         (c)2006 Japan Science and Tech Corp(JST)
File  99:Wilson Appl. Sci & Tech Abs 1983-2005/Dec
         (c) 2006 The HW Wilson Co.
File 111:TGG Natl.Newspaper Index(SM) 1979-2006/Jan 13
         (c) 2006 The Gale Group
File 144:Pascal 1973-2006/Dec W4
         (c) 2006 INIST/CNRS
File 239:Mathsci 1940-2005/Feb
         (c) 2005 American Mathematical Society
File 256:TecInfoSource 82-2005/Feb
         (c) 2005 Info.Sources Inc
```

23/3,K/1 (Item 1 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

03759854 E.I. No: EIP93111136658
Title: More efficient software **implementations of (generalized) DES**
Author: Pfitzmann, Andreas; Assmann, Ralf
Corporate Source: Universitat Hildesheim, Hildesheim, Ger
Source: Computers & Security v 12 n 5 Aug 1993. p 477-500
Publication Year: 1993
CODEN: CPSEDU ISSN: 0167-4048
Language: English

Title: More efficient software **implementations of (generalized) DES**
Abstract: This paper serves two purposes: we present some generalizations of the Data **Encryption** Standard (DES), and explain how to efficiently implement DES and its generalization in **software** . By preserving the macro structure of DES, but by allowing the user to choose (1...

...main memories, the big table is split into smaller ones that permute disjoint and compact **parts** of the input bits at the appropriate positions. To compute an entry in the big...

...G-DES, it does not seem to make sense to implement anything more narrow in **software** than G-DES with non-arbitrary E. Using these techniques, we get by far the fastest **software** implementations of DES (more specifically G-DES with non-arbitrary E) and G-DES known...

...and 24%. To avoid unnecessary IP and IP** minus **1 executions, and to enable multiple **encryption** in all modes of operation, our implementation supports multiple **encryption** . Our table implementation makes it possible to save key EORing (Exclusive OR equals bitwise addition...

...Memory requirements can be reduced by not copying bits which are input to one combined **S - box** . (Author abstract) 47 Refs.
Descriptors: ***Cryptography** ; Standards; Computer **software** ; Security of data; Data handling; Function evaluation; Computer programming languages
Identifiers: Authentication; **Software** implementation of Data **Encryption** Standard; Generalizations of DES; Modes of operation; Multiple **encryption** ; Implementation of permutations in assembly language; Concatenations

23/3,K/2 (Item 1 from file: 94)
DIALOG(R)File 94:JICST-EPlus

04690855 JICST ACCESSION NUMBER: 00A1007048 FILE SEGMENT: JICST-E
A Revised Nested SPN Cipher.
OKUMA KENJI (1); MURATANI HIROFUMI (1); MOTOYAMA MASAHIKO (1); KAWAMURA SHIN'ICHI (1); SANO FUMIHIKO (2)
(1) Toshiba Corp.; (2) Toshiba Sigikaise
Joho Shori Gakkai Kenkyu Hokoku, **2000** , VOL.2000,NO.80(CSEC-11), PAGE.37-42, TBL.7
JOURNAL NUMBER: Z0031BAO ISSN NO: 0919-6072
UNIVERSAL DECIMAL CLASSIFICATION: 621.391.037.3 681.3.02-759
LANGUAGE: Japanese COUNTRY OF PUBLICATION: Japan
DOCUMENT TYPE: Journal
ARTICLE TYPE: Original paper
MEDIA TYPE: Printed Publication

, 2000
...ABSTRACT: Type-I and Type-II based on a nested SPN strucuture where the upper-level **S - box** consists of the lower-level SP network hierarchically. The key scheduling **parts** are designed on a 256-bit modified Feistel structure. This paper proposes an improved version...
DESCRIPTORS: **cryptogram** ;
...BROADER DESCRIPTORS: **software** ;

23/3,K/3 (Item 2 from file: 94)
DIALOG(R)File 94:JICST-EPlus

01168848 JICST ACCESSION NUMBER: 91A0042670 FILE SEGMENT: JICST-E
Special issue on cryptography **and information security.**
Superdistribution: The concept and the architecture.
MORI R (1); KAWAHARA M (1)
(1) Univ. Tsukuba, Tsukuba-shi, JPN
Trans Inst Electron Inf Commun Eng E, **1990** , VOL.73,NO.7, PAGE.1133-1146, FIG.10, TBL.1, REF.11
JOURNAL NUMBER: F0699BBZ ISSN NO: 0387-236X
UNIVERSAL DECIMAL CLASSIFICATION: 681.3.02-759
LANGUAGE: English COUNTRY OF PUBLICATION: Japan
DOCUMENT TYPE: Journal
ARTICLE TYPE: Original paper
MEDIA TYPE: Printed Publication

Special issue on cryptography **and information security.**
Superdistribution: The concept and the architecture.
, **1990**
ABSTRACT: Superdistribution is an approach to distributing **software** in which **software** is made available freely and without restriction but is peotected from modifications and modes of usage not authorized by its vendor. By eliminating the need of **software** vendors to protect their products against piracy through copy protection and similar measures, superdistribution promotes unrestricted distribution of **software** . The superdistribution architecture we have developed provides three principal functions: administrative arrangements for collecting accounting information on **software** usage and fees for **software** usage; a accounting process that records and accumulates usage charges, payments, and the allocation of usage charges among **different software** vendors; and a defense mechanism, utilizing digitally protected modules, that protects the system against interference with its proper operation. Superdistribution **software** is distributed over public channels in encryped form. In order to participate in superdistribution, a computer must be equipped with an **S - box** - a digitally protected module containing microprocessors, RAM, ROM, and a real-time clock. The **S - box** preserves secret information such as a **decipheringkey** and manages the proprietary aspects of the superdistribution system. A **Software** Usage Monitor insures the intergrity of the system and keeps track of accounting information. The **S - box** can be realized as a digitally protected module in the form of a three-dimensional...
...DESCRIPTORS: **software** ; ...

... **cryptogram** ;
...BROADER DESCRIPTORS: electric apparatus and **parts** ; ...

... **parts** ;

23/3,K/4 (Item 1 from file: 144)
DIALOG(R)File 144:Pascal

13190643 PASCAL No.: 97-0454598
Improving linear cryptanalysis **of LOKI91 by Probabilistic counting method**
FSE '97 : fast software encryption : **Haifa, January 20-22, 1997**
SAKURAI K; FURUYA S
BIHAM Eli, ed
Department of Computer Science and Communication Engineering, Kyushu University, Hakozaki, Higashi-ku, Fukuoka 812-81, Japan
Fast software encryption. International workshop, 4 (Haifa ISR) 1997-01-20
Journal: Lecture notes in computer science, **1997** , 1267 114-133
Language: English

Improving linear cryptanalysis **of LOKI91 by Probabilistic counting method**
FSE '97 : fast software encryption : **Haifa, January 20-22, 1997**
1997
We improve linear **cryptanalysis** by introducing a technique of probabilistic counting into the maximum likelihood stage. In the original linear **cryptanalysis** based on maximum likelihood method with deterministic counting, the number of effective key and text bits is a multiple of the number of bit involved in the input to some **S - box** . Then, when larger **S - boxes** are used, 2R-method and even the 1R-methods can become impractical just because the...

... ciphers where 2R-method is impractical include LOKI91. We overcome this problem by selecting a **part** of the effective key bits and investigating the probabilistic behavior of the remained effective key...

... probability that the approximated formula with unknown inputs equals to zero. This extension of linear **cryptanalysis** make useful for 2R-attack on LOKI91, then improves the performance of previous attacks. Furthermore...

English Descriptors: **Software** engineering; Search algorithm; **Cryptography**

French Descriptors: Genie logiciel; Algorithme recherche; **Cryptographie**

23/3,K/5 (Item 1 from file: 239)
DIALOG(R)File 239:Mathsci

02766675 MR 98c#94018
On the difficulty of constructing cryptographically **strong** substitution boxes .
Zhang, Xian-Mo (Department of Mathematics, University of Wollongong, Wollongong, NSW 2500, Australia)
Zheng, Yuliang
Corporate Source Codes: 5-WLG
J.UCS
J.UCS. The Journal of Universal Computer Science, **1996** , 2, no. 3,

147--162 (electronic).
Language: English Summary Language: English
Subfile: MR (Mathematical Reviews) AMS
Abstract Length: LONG (48 lines)
Reviewer: Nyberg, Kaisa (Helsinki)

On the difficulty of constructing cryptographically **strong** substitution boxes .
1996 ,
...as an extended abstract [J. Seberry, X. M. Zhang and Y. Zheng, in Advances in **Cryptology** --- **CRYPTO** '94, (Santa Barbara, CA, 1994), 383--396, Lecture Notes in Comput. Sci., 839, Springer, Berlin, 1994].
Two significant recent advances in **cryptanalysis** , namely the differential attack put forward by E. Biham and A. Shamir [J. **Cryptology** 4 (1991), no. 1, 3--72; MR 93j:94020] and the linear attack by M. Matsui [in Advances in **Cryptology** ---EUROCRYPT '93, (Lofthus, 1993), 386--397, Lecture Notes in Comput. Sci. 765, Springer, Berlin, 1994] have had a crucial impact on the design of data **encryption** algorithms. In the paper under review a heuristic measure called ``robustness'' is proposed to be used for **substitution boxes** to measure resistance against **differential cryptanalysis** .
The **main part** of the paper consists of a study of a particular type of $n\times s$ **substitution boxes** which have a uniformly half-occupied difference distribution table (UHODDT). The authors consider **substitution boxes** with a UHODDT and balanced quadratic components particularly appealing. The main result of the paper is to prove that such **substitution boxes**0 do not exist if n or s is even. The examples given at the end ...

...that the assumption that all non-zero linear combinations of the output bits of the **substitution box** are balanced is essential. It is misleading not to include this assumption in the statement of Theorem 5.
Generalising the property of UHODDT the authors consider **substitution boxes** with two-valued difference distribution tables. The right-hand side of the first formula on...

...1)(2\sp n-1)$. This error was previously pointed out by Nyberg [in Fast **Software Encryption** , 111--130; per revr.], who also **first** proved Theorem 6 in a more general form. In this paper, the reviewer also gives...

...treatment of various synthesizing methods, which in the paper under review are considered only for **substitution boxes** with UHODDT.
Finally, it is shown that the difference distribution table of a differentially 2-uniform quadratic permutation embodies a Hadamard matrix.
No examples of **substitution boxes** with a two-valued but not half-occupied difference distribution table are given in this...

...have been readily available by Proposition 3 in a paper by Nyberg [in Advances in **cryptology** ---EUROCRYPT '93 (Lofthus, 1993), 55--64, Lecture Notes in Comput. Sci., 765, Springer, Berlin, 1994...
Descriptors: *94A60 -Information and communication, circuits- Communication, information- **Cryptography** (See also 11T71, 68P25)

23/3,K/6 (Item 2 from file: 239)
DIALOG(R)File 239:Mathsci

02036446 MR 88h#94004
Advances in cryptology --- CRYPTO **'86.**

Proceedings of the conference on the theory and applications of **cryptographic** techniques held at the University of California, Santa Barbara, Calif., August 11--15, 1986. Edited by A. M. Odlyzko.
Contributors: Odlyzko, A. M.
Publ: Springer-Verlag, Berlin-New York,
1987 , xii+489 pp. ISBN: 3-540-18047-8
Series: Lecture Notes in Computer Science, 263.
Language: English
Advances in **cryptology** --- **CRYPTO** '86; Conference: Theory and applications of **cryptographic** techniques; Santa Barbara, Calif.,; Lecture Notes in Computer Science, 1986 263
Subfile: MR (Mathematical Reviews) AMS
Abstract Length: LONG (72 lines)
Reviewer: Editors

Advances in cryptology --- CRYPTO **'86.**
Proceedings of the conference on the theory and applications of **cryptographic** techniques held at the University of California, Santa Barbara, Calif., August 11--15, 1986. Edited...
1987 ,
Advances in **cryptology** --- **CRYPTO** '86; Conference: Theory and applications of **cryptographic** techniques; Santa Barbara, Calif.,; Lecture Notes in Computer Science,
\{**CRYP** TO '85 has been reviewed [MR 87d:94002].\}\
Contents:\ E. F. Brickell, J. H. Moore [Judy Hennessey Moore] and M. R. Purtill, Structure in the $ **S** $- **boxes** of the DES (extended abstract) (pp. 3--8); Judy H. Moore and Gustavus J. Simmons...

...pp. 9--32); T. R. N. Rao and Kil-Hyun Nam, Private-key algebraic-coded **cryptosystems** (pp. 35--48); Wiebren de Jonge and David Chaum, Some variations on RSA signatures & their...

...Avi Wigderson, How to prove all NP statements in zero-knowledge and a methodology of **cryptographic** protocol design (extended abstract) (pp. 171--185); Amos Fiat and Adi Shamir, How to prove...

...Demonstrating possession of a discrete logarithm without revealing it (pp.\ 200--212); Josh Cohen Benaloh, **Cryptographic** capsules: a disjunctive primitive for interactive protocols (pp. 213--222); Gilles Brassard and Claude Crepeau...
...P. A. Scott, L. E. Peppard and S. E. Tavares, VLSI implementation of public-key **encryption** algorithms (pp. 277--301); T. Beth, B. M. Cook and D. Gollmann, Architectures for exponentiation...

...sp n)$ (pp. 302--310); Paul Barrett, Implementing the Rivest Shamir and Adleman public key **encryption** algorithm on a standard digital signal processor (pp. 311--323); Robert R. Jueneman, A high...

...347--377); Silvio Micali, Charles Rackoff and Bob Sloan, The notion of security for probabilistic **cryptosystems** (extended abstract) (pp. 381--392); Neal R. Wagner, Paul S. Putter and Marianne R. Cain...

...constructions and bounds for authentication codes (pp.\ 418--425); Oded Goldreich, Towards a theory of **software** protection (extended abstract) (pp. 426--439).
Pierre Beauchemin, Gilles Brassard, Claude Crepeau and Claude Goutier...

...Matyas, Public key registration (pp. 451--458); Yvo Desmedt, Is there an ultimate use of **cryptography** ? (extended abstract) (pp. 459--463); Louis C. Guillou and Michel Ugon, Smart card, a highly...

Descriptors: ...; Computer science (For papers involving machine computations and programs in a specific mathematical area, see **section** --04 in that area)-Proceedings, conferences, etc...

...94A60 -Information and communication, circuits-Communication, information- **Cryptography** (See also 11T71)

28/3,K/7 (Item 7 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014368897 **Image available**
WPI Acc No: 2002-189599/200225
XRPX Acc No: N02-143687
Substitution **permission network structure type** Pseudo **matrix computing apparatus outputs value indicating existence probability of linear converting unit corresponding to** S boxes
Patent Assignee: FUJITSU LTD (FUIT); ITO K (ITOK-I); SHIMOYAMA T (SHIM-I); TAKENAKA M (TAKE-I); TORII N (TORI-I); YAJIMA J (YAJI-I); YANAMI H (YANA-I); YOKOYAMA K (YOKO-I)
Inventor: ITO K; SHIMOYAMA T; TAKENAKA M; TORII N; YAJIMA J; YANAMI H; YOKOYAMA K
Number of Countries: 028 Number of Patents: 004
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
EP 1172963	A2	20020116	EP 2001302883	A	20010328	200225 B
US 20020021801	A1	20020221	US 2001813024	A	20010321	200225
JP 2002091295	A	20020327	JP 2001207327	A	20010709	200225
JP 2002091297	A	20020327	JP 2001207326	A	20010709	200225

Priority Applications (No Type Date): JP 2000212814 A 20000713; JP 2000212813 A 20000713
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 1172963	A2	E	46	H04L-009/06	
Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI TR					
US 20020021801	A1			H04L-009/06	
JP 2002091295	A		13	G09C-001/00	
JP 2002091297	A		17	G09C-001/00	

Substitution **permission network structure type** Pseudo **matrix computing apparatus outputs value indicating existence probability of linear converting unit corresponding to** S boxes

Abstract (Basic):
... Computer (1) receives unequally **divided** bit numbers and an output unit (3) outputs a value AT' indicating the existence probability of an appropriate linear converting unit corresponding to **S boxes** having **divided** bits as their input and output. When output value is positive, it is determined that...

...present and a psuedo MDS matrix is obtained corresponding to a MDS matrix having unequally **divided** bits.
... a) **Pseudo** matrix computation method...
...b) computer readable recorded medium storing **pseudo** matrix computation program...

...For computation of **pseudo** MDS matrix using **substitution** permission network (SPN) structure in combination with Feistel structure...

...The figure shows the block diagram of **pseudo** matrix computing apparatus...
Title Terms: **SUBSTITUTE** ;
...International Patent Class (Main): **H04L-009/06**
Manual Codes (EPI/S-X): **W01-A06C1**

US 20020021801A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2002/0021801 A1

Shimoyama et al. (43) Pub. Date: Feb. 21, 2002

(54) COMPUTING APPARATUS USING AN SPN STRUCTURE IN AN F FUNCTION AND A COMPUTATION METHOD THEREOF

(76) Inventors: Takeshi Shimoyama, Kawasaki (JP); Koichi Ito, Kawasaki (JP); Masahiko Takenaka, Kawasaki (JP); Naoya Torii, Kawasaki (JP); Jun Yajima, Kawasaki (JP); Hitoshi Yanami, Kawasaki (JP); Kazuhiro Yokoyama, Kawasaki (JP)

Correspondence Address:
STAAS & HALSEY LLP
700 11TH STREET, NW
SUITE 500
WASHINGTON, DC 20001 (US)

(21) Appl. No.: 09/813,024

(22) Filed: Mar. 21, 2001

(30) Foreign Application Priority Data

Jul. 13, 2000 (JP) 2000-212813

Jul. 13, 2000 (JP) 2000-212814

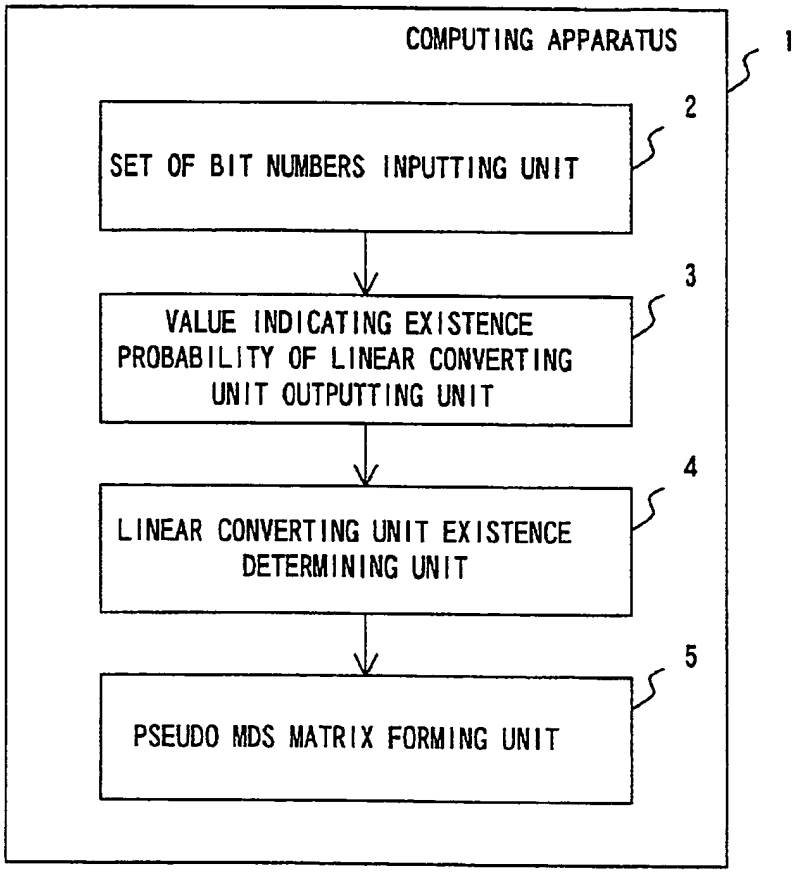
Publication Classification

(51) Int. Cl.[7] H04L 9/06

(52) U.S. Cl. 380/29; 380/37

(57) ABSTRACT

By providing a unit receiving the input of a set T of bit numbers that are obtained by unequally dividing all the bit numbers of input data to be given to a computing apparatus, a unit outputting a value A_T indicating an existence probability of an appropriate linear converting unit corresponding to a plurality of S boxes of which the input and output bit numbers are equivalent to the divided bit numbers, a unit determining that an appropriate linear converting unit is present when the value of A_T is positive, and a unit forming a pseudo MDS matrix as the linear converting unit, computation is executed using a unit with an excellent data diffusion performance as the linear converting unit in SPN structure, when the input number is not the same as the output number among a plurality of S boxes of the SPN structure in an F function.



What is claimed is:

1. A computing apparatus using SPN structure having a plurality of S boxes and a linear converting unit in an F function, comprising:

a set of bit numbers inputting unit receiving an input of a set $T=\{t_1, t_2, t_3 \ldots t_r\}$ of bit numbers obtained by unequally dividing all bit numbers of input data to be given to the computing apparatus; and

a value indicating existence probability of linear converting unit outputting unit outputting a value A_T indicating an existence probability of an appropriate linear converting unit corresponding to a plurality of S boxes of which input and output bit numbers are equivalent to the divided bit numbers.

2. The computing apparatus according to claim 1, wherein said value indicating existence probability of linear converting unit outputting unit comprises a minimum value determining unit obtaining a minimum value $u_k(k=1, 2, \ldots, r)$ of a sum of elements of a set formed by selecting optional k elements from elements of the set T, and a maximum value determining unit obtaining amaximum value $v_k(k=1, 2, 3, \ldots, r)$ of a sum of elements of a set formed by selecting optional k elements from elements of the set T, wherein

a value obtained by subtracting a maximum value of k' that satisfies $u_k \geqq v_{k'}(k'=0, 1, \ldots, r, v_0=0)$ for a value k, from k is set as $w_k(k=1, 2, \ldots, r)$, and the value A_T is obtained by subtracting a maximum value of w_k from a value of (r+1).

3. The computing apparatus according to claim 1, further comprising:

a linear converting unit existence determining unit determining whether the vale A_T is positive, and determining that the appropriate linear converting unit is present when the value is positive.

4. The computing apparatus according to claim 2, further comprising:

a linear converting unit existence determining unit determining whether the value A_T is positive, and determining that the appropriate linear converting unit is present when the value is positive.

5. The computing apparatus according to claim 3, further comprising:

a pseudo MDS matrix forming unit forming as the linear converting unit, a pseudo MDS matrix corresponding to an MDS matrix in a case where the bits are unequally divided when it is determined that the linear converting unit is present.

6. The computing apparatus according to claim 4, further comprising:

a pseudo MDS matrix forming unit forming as the linear converting unit, a pseudo MDS matrix corresponding to an MDS matrix in a case where the bits are unequally divided when it is determined that the linear converting unit is present.

7. The computing apparatus according to claim 5, wherein the pseudo MDS matrix forming unit sets a matrix M of r columns and r rows to $M=(M_{ij})$ $(i=1, 2, \ldots, r, j=1, 2, \ldots, r)$ while setting as an element a partial matrix M_{ij} of t_i columns and t_j rows of which an element is 0 or 1, obtains $c(e)=e+r-A_T+1$ for each positive number from e=1 to (A_T-1), obtains a set $T_1=\{t_{i1}, t_{i2}, \ldots, t_{ie}\}$ formed by optionally selecting e elements from elements of the set T and a set $T_2=\{t_{j1}, t_{j2}, \ldots, t_{jc(e)}\}$ formed by optionally selecting c(e) elements from elements of the set T, and obtains a matrix M such that a value of a small matrix of an optional matrix M corresponding to the set (T_1, T_2) and a value of a rank of a small matrix of an optional matrix M corresponding to the set (T_2, T_1) is equal to either a column number of a small matrix of the matrix M or a number of ranks of a small matrix of a matrix M.

8. The computing apparatus according to claim 5, wherein the pseudo MDS matrix forming unit sets a matrix M of r columns and r rows to $M=(M_{ij})$ $(i=1, 2, \ldots, r, j=1, 2, \ldots, r)$ while setting as an element a partial matrix M_{ij} of t_i columns and t_j rows of which an element is 0 or 1, obtains $c(e)=e+r-A_T+1$ for each positive number from e=1 to (A_T-1), obtains a set $T_1=\{t_{i1}, t_{i2}, \ldots, t_{ie}\}$ formed by optionally selecting e elements from elements of the set T and a set $T_2=\{t_{j1}, t_{j2}, \ldots, t_{jc(e)}\}$ formed by optionally selecting c (e) elements from elements of the set T, and obtains a matrix M such that a value of a small matrix of an optional matrix M corresponding to the set (T_1, T_2) and a value of a rank of a small matrix of an optional matrix M corresponding to the set (T_2, T_1) is equal to either a column number of a small matrix of the matrix M or a number of ranks of a small matrix of a matrix M.

9. The computing apparatus according to claim 7, wherein a small matrix corresponding to the sets (T_1, T_2) is configured by a partial matrix designated by columns respectively corresponding to the $t_{i1}, t_{i2}, \ldots, t_{ie}$ and rows respectively corresponding to the $t_{j1}, t_{j2}, \ldots, t_{jc(e)}$ among partial matrixes M_{ij} that function as elements of the r columns and r rows to configure the matrix $M=(M_{ij})$.

10. The computing apparatus according to claim 8, wherein a small matrix corresponding to the sets (T_1, T_2) is configured by a partial matrix designated by columns respectively corresponding to the $t_{i1}, t_{i2}, \ldots, t_{ie}$ and rows respectively corresponding to the $t_{j1}, t_{j2}, \ldots, t_{jc(e)}$, among partial matrixes M_{ij} that function as elements of the r columns and r rows to configure the matrix $M=(M_{ij})$.

11. A computation method using SPN structure having a plurality of S boxes and a linear converting unit in an F function, comprising:

receiving an input of a set $T=\{t_1, t_2, t_3 \ldots t_r\}$ of bit numbers obtained by unequally dividing all bit numbers of input data to be given; and

outputting a value A_T indicating an existence probability of an appropriate linear converting unit corresponding to a plurality of S boxes of which input and output bit numbers are equivalent to the divided bit numbers.

12. The computation method using SPN structure having an F function according to claim 7, comprising:

determining whether the vale A_T is positive or not; and

determining that the appropriate linear converting unit is present when the value is positive.

13. The computation method according to claim 12, wherein a pseudo MDS matrix corresponding to an MDS matrix in a case where the bits are equally divided is formed as the linear converting unit.

14. A computer-readable portable recording medium used by a computer executing a computation process using SPN structure having a plurality of S boxes and a linear convert-

ing unit in an F function, storing a program for causing the computer to perform, comprising:

receiving an input of a set $T=\{t_1, t_2, t_3, \ldots t_r\}$ of bit numbers obtained by unequally dividing all bit numbers of input data to be given; and

outputting a value A_T indicating an existence probability of an appropriate linear converting unit corresponding to a plurality of S boxes of which input and output bit numbers are equivalent to the divided bit numbers.

15. A computing apparatus in which Feistel structure and SPN structure are combined, receiving data input and setting a computation result for the data input as a data output, wherein

at least one first data converting units that perform data conversion using the Feistel structure, and at least one second data converting units that perform data conversion using the SPN structure are continuously combined between the data input and the data out.

16. The computing apparatus according to claim 15, wherein the SPN structure comprises a nonlinear converting unit having an input/output bit number obtained by dividing a block length of one block of the data input by a word length, and a liner converting unit that uses interleaving conversion.

17. The computing apparatus according to claim 15, comprising:

a nonlinear converting unit having a probability 0 that for a set of input data in which a differential appears only on at least one fixed input bit among input bits to the nonlinear converting unit, a differential appears for a set of output data in which a differential appears on at least one fixed output bits located at the same location as at least one fixed input bits, and further a probability 1/2 that an optional linear relational equation only related to at least one fixed output bits and at least one fixed output bits, realizes between all the input data and output data 1/2, is provided, as a nonlinear converting unit configuring the SPN structure.

18. The computing apparatus according to claim 16, comprising:

a nonlinear converting unit having a probability 0 that for a set of input data in which a differential appears only on at least one fixed input bit among input bits to the nonlinear converting unit, a differential appears for a set of output data in which a differential appears on at least one fixed output bits located at the same location as at least one fixed input bits, and further a probability 1/2 that an optional linear relational equation only related to at least one fixed output bits and at least one fixed output bits, realizes between all the input data and output data 1/2, is provided, as a nonlinear converting unit configuring the SPN structure.

19. A computation method in which Feistel structure and SPN structure are combined, receiving a data input and setting a computation result for the data input as a data output, wherein

at least one piece of first data conversion that performs data conversion using the Feistel structure and at least one piece of second data conversion that performs data conversion using the SPN structure are combined to be executed between the data input and the data output.

20. The computation method in which the Feistel structure and the SPN structure are combined according to claim 19, wherein

in first data conversion using the SPN structure, nonlinear conversion of which a number of input bits and a number of output bits are equivalent to a value obtained by dividing a block length of one block of a data input by a word length, and

liner conversion that uses interleaving conversion, are executed.

21. The computing method in which the Feistel structure and the SPN structure are combined according to claim 19, wherein

nonlinear conversion having a probability 0 that for a set of input data in which a differential appears only on at least one fixed input bit among input bits to be used for the nonlinear conversion, a differential appears for a set of output data in which a differential appears on at least one fixed output bits located at the same location as the at least one fixed input bits, and further having a probability 1/2 that an optional linear relational equation only related to the at least one fixed input bits and the at least one fixed output bits is realized between all the input data and output data, is executed as nonlinear conversion to be executed in the SPN structure.

22. The computing method in which the Feistel structure and the SPN structure are combined according to claim 20, wherein

nonlinear conversion having a probability 0 that for a set of input data in which a differential appears only on at least one fixed input bit among input bits to be used for the nonlinear conversion, a differential appears for a set of output data in which a differential appears on at least one fixed output bits located at the same location as the at least one fixed input bits, and further having a probability 1/2 that an optional linear relational equation only related to the at least one fixed input bits and the at least one fixed output bits is realized between all the input data and output data, is executed as nonlinear conversion to be executed in the SPN structure.

23. A portable computer-readable recording medium being used for a computer that executes computation of receiving data input and that sets a computation result for the input data as a data output, and storing a program causing the computer to perform, comprising:

combining and executing at least one piece of first data conversion that performs data conversion using Feistel structure; and at least one piece of second data conversion that performs data conversion using SPN structure between the data input and the data output.

24. A computing apparatus using SPN structure having a plurality of S boxes and a linear converting unit in an F function, comprising:

set of bit numbers inputting means for receiving an input of a set $T=\{t_1, t_2, t_3 \ldots t_r\}$ of bit numbers obtained by unequally dividing all bit numbers of input data to be given to the computing apparatus; an

value indicating existence probability of linear converting unit outputting means for outputting a value A_T indicating an existence probability of an appropriate linear converting unit corresponding to a plurality of S boxes

of which input and output bit numbers are equivalent to the divided bit numbers.

25. A computing apparatus in which Feistel structure and SPN structure are combined, for receiving a data input, and setting a computation result for the data input as a data output, comprising:

at least one first data converting means for performing data conversion using the Feistel structure; and

at least one second data converting means for performing data conversion using the SPN structure,

wherein said first data converting means and said second data converting means are continuously combined between the data input and the data output.

* * * * *

28/3,K/42 (Item 42 from file: 350)
DIALOG(R)File 350:Derwent WPIX
(c) 2006 Thomson Derwent. All rts. reserv.

012783227 **Image available**
WPI Acc No: 1999-589453/199950
XRPX Acc No: N99-434625

Block cipher implementing method for encryption **and message authentication in microprocessor and microcontroller controlled devices**

Patent Assignee: MICROSOFT CORP (MICT)
Inventor: YUVAL G A
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5956405	A	19990921	US 97784841	A	19970117	199950	B

Priority Applications (No Type Date): US 97784841 A 19970117
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5956405	A		23	H04K-001/00	

Block cipher implementing method for encryption **and message authentication in microprocessor and microcontroller controlled devices**

Abstract (Basic):

... Block of application code included in ROM is selected by microprocessor, so as to perform **initial** operation. The selected blocks are used as a **substitution box** . Cipher operation on set of bits, is performed by accessing **portion** of selected block of application code to obtain set of **substitution** bits. The set of **substitution** bits are then **substituted** for the **portion** of set of bits.

... For implementing **encryption** and message authentication in microprocessor and microcontroller controlled device such as home appliances...

...By performing large number of **substitution** operation and data manipulation operations as **part** of the block cipher, high degree of security can be achieved without use of dedicated **S - box** .

...Title Terms: **ENCRYPTION** ;
International Patent Class (Main): **H04K-001/00**
Manual Codes (EPI/S-X): **T01-D01** ...

... **W01-A05A** ...

... **W01-A05B**

US005956405A

United States Patent [19]

Yuval

[11] Patent Number: 5,956,405

[45] Date of Patent: Sep. 21, 1999

[54] IMPLEMENTATION EFFICIENT ENCRYPTION AND MESSAGE AUTHENTICATION

[75] Inventor: Gideon A. Yuval, Mercer Island, Wash.

[73] Assignee: Microsoft Corporation, Redmond, Wash.

[21] Appl. No.: 08/784,841

[22] Filed: Jan. 17, 1997

[51] Int. Cl.[6] H04K 1/00

[52] U.S. Cl. 380/29

[58] Field of Search 380/29

[56] References Cited

U.S. PATENT DOCUMENTS

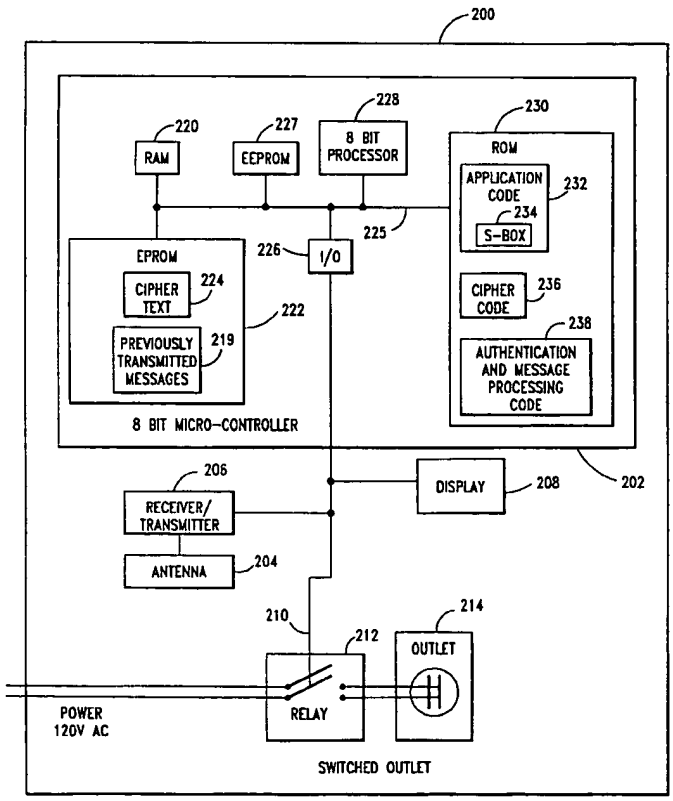
3,962,539	6/1976	Ehrsam et al.	380/29
4,074,066	2/1978	Ehrsam et al.	380/29
4,172,213	10/1979	Barnes et al.	380/29
4,274,085	6/1981	Marino, Jr.	380/29

Primary Examiner—Salvatore Cangialosi
Attorney, Agent, or Firm—Michaelson & Wallace; Peter L. Michaelson; Michael P. Straub

[57] ABSTRACT

Encryption and authentication techniques which can be implemented on inexpensive, e.g., 8-bit, microprocessors and micro-controllers, using very little of the microprocessor's memory, are described. While the described techniques require little system resources to implement they still provide a good degree of security. In accordance with the present invention, in order to avoid having to specifically dedicate a portion of the microprocessor's limited memory for use as a substitution box, a portion of the code stored in the microprocessor's memory, dedicated to performing another function, is selected to serve as an S-box. This memory saving technique is used to implement a block cipher. The block cipher is used in combination with a series of other data manipulation operations, including XOR operations and rotate operations, to provide a good degree of system security. The operations used to implement the techniques of the present invention are capable of being implemented using 8 bit instructions making the techniques of the present invention well suited for implementation on 8 bit systems such as those used in home and auto control applications. The message protocol and encryption scheme of the present invention involves the subtracting of current message payloads from previously received message payloads to distinguish between new messages and repeated messages which have already been acted upon. Messages are acted upon only once thereby rendering the recording and playing back of previous messages ineffective at defeating system security.

26 Claims, 14 Drawing Sheets



message, and/or comparing the unencrypted message payload to a list of commands or instructions stored in memory to determine if a valid message has been received.

As a result of the message subtraction feature of the present invention, an unencrypted message payload value of zero results if the current message is a repeat of the last message. Repeated messages may be the result of the transmitting device or controller failing to receive an acknowledgment signal, e.g., due to noise problems. If the unencrypted message payload **952** is determined to have a value of zero in step **924**, operation progresses to step **936** wherein a message acknowledgment signal is sent to the transmitting device. Operation then progresses to step **940**, without any further action being taken in response to the message payload, wherein the system returns to step **904** to await receipt of the next message.

If, in step **924**, e.g., through the use of an instruction or command look-up operation, it is determined that the unencrypted message payload represents a valid instruction or command, system operation progresses to step **932**. In step **932** a message acknowledgment signal is transmitted to the source of the message. The instruction or command is then acted upon in step **934**, e.g., the relay **212** of the device **200** is controlled by the micro-controller **202** in response to the received message to switch from one position to another. The encrypted message payload **542**, of the message which was just acted upon, is then stored in step **935** for use in decoding future messages from the same source as the current message. The encrypted message payload **542** is stored in section **219** of EPROM **222** which has a dedicated memory space for each potential message source. Once the encrypted message payload is stored, operation progresses to step **940** and the system returns to step **904** to await receipt of another message.

If, it is determined in step **924**, that an unencrypted message is invalid, e.g., because it does not correspond to a valid command or instruction, operation progresses from step **924** to step **926**. In step **926** an invalid message counter implemented in, e.g., the EPROM **222**, is incremented to reflect receipt of the current invalid message. In step **928** a check is made to determine if the counted number of invalid messages exceeds a preselected threshold value. If the preselected threshold is not exceeded, operation progresses to step **940** which returns system operation to step **904** to await the receipt of another message.

However, if, in step **928**, it is determined that the number of counted erroneous messages exceeds a threshold value, operation progresses to step **930**. Exceeding the preselected threshold value of erroneous messages is indicative of an attempt by an unauthorized individual to penetrate the home control system's security features. In order to warn of the potential threat to system security, in step **930** an alarm message is transmitted to the controller **102**.

From step **930** operation progresses to step **940** which returns system operation to step **904** to await the receipt of another message.

In accordance with the present invention, the value of the counter maintained in step **928** may be reset at periodic intervals to take into consideration invalid messages resulting from, e.g., signal noise or interference.

While the present invention has been illustrated with reference to an exemplary embodiment, those skilled in the art will appreciate that various changes in form and detail may be made without departing from the intended scope of the present invention as defined in the appended claims. For example, a longer message length such as 16 or 32 byte messages may be used. Because of the variations that can be applied to the illustrated and described embodiment of the invention, the invention should be defined solely with reference to the appended claims.

What is claimed is:

1. A method of implementing a block cipher in a first device, comprising the steps of:

selecting a block of application code used by the device in performing a first operation for use as a substitution box; and

performing a cipher operation on a first set of bits, the cipher operation being different than the first operation and including the steps of:

accessing a portion of the selected block of application code to obtain a set of substitution bits therefrom; and

substituting the set of substitution bits for a portion of the first set of bits.

2. The method of claim **1**, wherein the step of performing a cipher operation on the first set of bits further includes the step of:

rotating the first set of bits.

3. The method of claim **2**, wherein the step of performing a cipher operation on the first set of bits further includes the step of:

repeatedly performing the accessing, substituting and rotating steps.

4. The method of claim **3**, wherein the step of selecting a block of application code includes the step of:

compressing a plurality of application code blocks and selecting the application code block which compresses the least for use as the substitution box.

5. The method of claim **3**, wherein the step of selecting a block of application code includes the step of:

selecting a block of code which is common to a plurality of devices which communicate with the first device.

6. A method of implementing a system including a plurality of devices, each device including a processor and memory, the memory of each device including application code for use in performing a first function, the method comprising the steps of:

identifying application code that is included in each one of the plurality of devices;

selecting a block of the identified application code to serve the function of being a substitution box, the substitution box function being different from the first function.

7. The method of claim **6**, further comprising the step of programming each of the plurality of devices with code for implementing a block cipher operation.

8. The method of claim **7**, wherein the step of programming each device to perform a block cipher operation includes the step of:

programming each device to perform a plurality of substitution and shift operations as part of the block cipher operation.

9. The method of claim **8**, further comprising the step of programming each of the plurality of devices to:

receive a first encrypted message;

store at least a portion of the first received encrypted message; and

upon receiving a subsequent encrypted message subtract at least the stored portion of the first encrypted message from at least a portion of the subsequently received encrypted message to thereby determine if the subse-

quent encrypted message is a repeat of the first encrypted message.

10. The method of claim 8, further comprising the step of:

assigning each of the plurality of devices a unique identifier;

storing the unique identifier assigned to each device in the device to which the identifier is assigned; and

storing within each particular one of the plurality of devices, the unique identifier assigned to the devices with which the particular device in which the unique identifiers are being stored is capable of communicating.

11. The method of claim 10, further comprising the step of;

programming each of the plurality of devices with an identical code to be used when performing a cipher operation.

12. The method of claim 10, further comprising the step of programming each of the plurality of devices to:

receive a first encrypted message;

store at least a portion of the first received encrypted message; and

upon receiving a subsequent encrypted message subtract at least the stored portion of the first encrypted message from at least a portion of the subsequently received encrypted message to thereby determine if the subsequent encrypted message is a repeat of the first encrypted message.

13. The method of claim 12, wherein the each of the plurality of devices is programmed to implement only 8 bit operations.

14. A system, comprising:

a first device, the first device including:

a first processor;

first non-volatile memory coupled to the microprocessor;

first application code stored in the non-volatile memory; and

first means for implementing a block cipher using a segment of the first application code stored in the first non-volatile memory as a first substitution box.

15. The system of claim 14, further comprising:

a second device, the second device including:

a second processor;

second non-volatile memory coupled to the microprocessor;

second application code stored in the second non-volatile memory; and

second means for implementing a block cipher using a segment of the second application code stored in the second non-volatile memory as a second substitution box, and wherein the segments of the first and second application code used for the first and second substitution boxes have an identical content.

16. The system of claim 15, wherein each of the first and second devices further include:

means for storing received messages; and

means for comparing a stored message to a received message to determine if the received message is a repeat of a stored message.

17. The system of claim 15, wherein the first device further includes:

a first authentication module for generating a message authenticator as a function of a message to be transmitted, the implemented block cipher, and a stored code common to both the first and second devices.

18. The system of claim 17, wherein the second device further includes:

a second authentication module for generating a message authenticator as a function of a received message, the implemented block cipher and the stored code common to both the first and second devices.

19. The system of claim 18, wherein the first authentication module includes means for logically XORing preselected portions of the message to be transmitted together to generate a portion of a first cipher key.

20. The system of claim 19, wherein the second authentication module includes means for logically XORing preselected portions of the received message together to generate a portion of a second cipher key.

21. The system of claim 19, wherein the means for implementing a block cipher included in each of the first and second devices includes means for rotating data and performing a plurality of substitution operations as a function of the content of the first and second substitution boxes, respectively.

22. The system of claim 19, wherein each of the first and second devices further include:

means for storing received messages; and

means for comparing a stored message to a received message to determine if the received message is a repeat of a stored message.

23. The method of claim 15, wherein the means for implementing a block cipher included in each of the first and second devices includes means for rotating data and performing a plurality of substitution operations as a function of the content of the first and second substitution boxes, respectively.

24. The system of claim 15,

wherein the first device is a controller which further includes an antenna for transmitting messages to the second device; and

wherein the second device includes an antenna for receiving messages from the first device.

25. The system of claim 15,

wherein the first device is a home appliance controller; and

wherein the second device is a remotely controlled switched outlet.

26. A computer readable medium comprising:

computer executable instructions for performing the steps of:

selecting a block of application code used by the device in performing a first operation for use as a substitution box; and

performing a cipher operation, the cipher operation being different from the first operation, on a first set of bits, the cipher operation including the steps of:

accessing a portion of the selected block of application code to obtain a set of substitution bits therefrom; and

substituting the set of substitution bits for a portion of the first set of bits.

* * * * *

28/3,K/37 (Item 37 from file: 350)
DIALOG(R)File 350:Derwent WPIX

013051477 **Image available**
WPI Acc No: 2000-223331/200019
XRPX Acc No: N00-167363
Practical S - box design used in cryptographic system and digital data processing
Patent Assignee: ENTRUST TECHNOLOGIES LTD (ENTR-N)
Inventor: ADAMS C M; MISTER S J M
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 6031911	A	20000229	US 9621983	A	19960718	200019 B
			US 97895875	A	19970717	

Priority Applications (No Type Date): US 9621983 P 19960718; US 97895875 A 19970717
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6031911	A		15	H04K-001/00	Provisional application US 9621983

Practical S - box design used in cryptographic system and digital data processing
Abstract (Basic):
... The method involves generating the **S - box** which is used in ciphering digital data. The **S - box** is capable of receiving m inputs and producing n outputs, where m and n are positive integers greater than 1. Ciphering of digital data is performed using the **first** generated **S - box** with n outputs.
... In generating the **S - box** , the desired operation characteristics for the **S - box** are **first** selected. The **first S - box** with m inputs and k outputs is then generated, where k is less tan n...

...the Boolean function of the m inputs is also generated. The operational characteristics of a **temporary** formed **S - box** are determined by the **first S - box** and the column. The **first S - box** is then modified by appending the column, if the **temporary S - box** possesses the desired operational characteristics. The said steps are repeated until the **first S - box** has n outputs...

...An INDEPENDENT CLAIM is also included for the digital data ciphering system using **S - box** .
...
...Used in **cry**ptographic system and digital data processing. For constructing large **substitution boxes** used in improving the security of symmetric block ciphers...

...Increases security of existing ciphers that uses **S - boxes** by constructing **S - boxes** with ideal characteristics rather than using randomly generated **S - boxes** . Constructs **S - box** that is resistant to known **cryptanalytic** attacks. Produces several **S - boxes** of ideal characteristics in reduced processing time...

...The figure shows the system for executing the practical **S - box** design
...Title Terms: **CRYPTOGRAPHIC** ;
International Patent Class (Main): **H04K-001/00**
Manual Codes (EPI/S-X): **T01-D01** ...

... T01-J12C ...

... W01-A05A ...

... W01-C08F

US006031911A

United States Patent [19]

Adams et al.

[11] Patent Number: 6,031,911

[45] Date of Patent: Feb. 29, 2000

[54] PRACTICAL S BOX DESIGN

[75] Inventors: Carlisle M. Adams, Ottawa; Serge J. M. Mister, Amherstview, both of Canada

[73] Assignee: Entrust Technologies, Ltd., Ottawa, Canada

[21] Appl. No.: 08/895,875

[22] Filed: Jul. 17, 1997

Related U.S. Application Data

[60] Provisional application No. 60/021,983, Jul. 18, 1996.

[51] Int. Cl.[7] H04K 1/00

[52] U.S. Cl. 380/29; 380/37; 380/46

[58] Field of Search 380/29, 37, 46

[56] References Cited

U.S. PATENT DOCUMENTS

5,003,597	3/1991	Merkle .	
5,317,639	5/1994	Mittenthal	380/37
5,351,299	9/1994	Matsuzaki et al. .	
5,511,123	4/1996	Adams	380/29
5,778,074	7/1998	Garcken et al.	380/37
5,796,837	8/1998	Kim et al.	380/28

OTHER PUBLICATIONS

A. Youseff, On the Linear Approximation of Injective S-boxes, Department of Electrical and Computer Engineering Queen's University, Kingston, Ontario, Canada, pp. 1–5.

A. Youseff, S. Tavares, S. Mister, C. Adams, Linear Approximation of Injective S-Boxes, Electronics Letters, vol. 31, Dec. 1995, pp. 2165–2166.

C. M. Adams, Constructing Symmetric Ciphers Using the CAST Design Procedure, Designs, Codes and Cryptography, Kluwer Academic Publishers, Boston, 1997, pp. 283–316.

C. M. Adams, Designing DES-Like Ciphers with Guaranteed Resistance to Differential and Linear Attacks, Workshop Record of the Workshop on Selected Areas in Cryptography (SAC 95), May 18–19, 1995, pp. 133–144.

C. M. Adams and S. E. Tavares, Generating and Counting Binary Bent Sequences, IEEE Transactions on Information Theory, vol. IT-36, 1990, pp. 1170–1173.

E. Biham, On Matsui's Linear Cryptanalysis, Advances in Cryptology—Proceedings of Eurocrypt '94, Springer-Verlag, Berlin, 1995, pp. 341–355.

E. Biham and A. Shamir, Differential Cryptanalysis of DES-Like Cryptosystems, Advances in Cryptology: Proceedings of CRYPTO '90, Springer-Verlag, Berlin, 1991, pp. 1–21.

J. F. Dillon, A Survey of Bent Functions, NSA Technical Journal, Special Issue, 1972, pp. 191–215.

K. Nyberg, Perfect Nonlinear S-Boxes. Advances in Cryptology—Proceedings of Eurocrypt '91, Springer-Verlag, Berlin, 1991, pp. 378–385.

B. Preneel, W. Van Leekwijck, L. Van Linden, R. Govaerts, and J. Vandewalle, Propagation characteristics of boolean functions, Advances in Cryptology: Proceedings of Eurocrypt '90, Springer-Verlag, Berlin, 1991, pp. 161–173.

A. F. Webster and S. E. Tavares, On the Design of S-Boxes, Advances in Cryptology; Proceedings of CRYPTO '85, Springer-Verlag, New York, 1986, pp. 523–534.

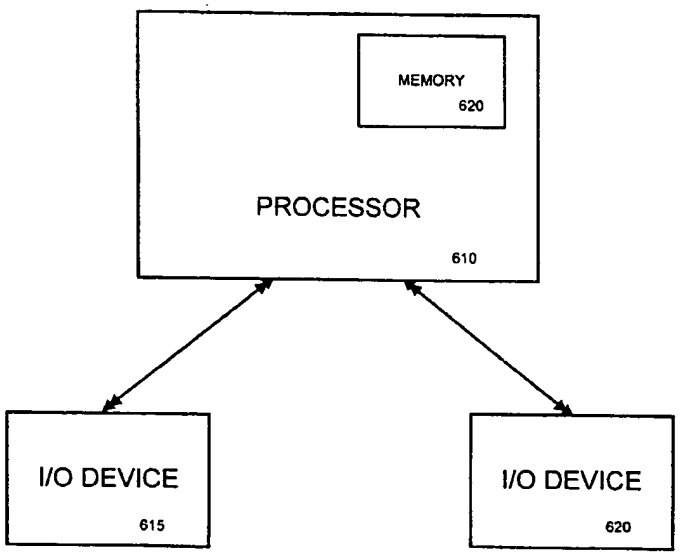
Primary Examiner—David Cain

Attorney, Agent, or Firm—Markison & Reckamp, P.C.

[57] ABSTRACT

A method of generating a substitution box (S-box) involves generating an S-box with desired characteristics, forming a new S-box with another column such that the new S-box has the desired characteristics as well, and continuing to add columns in these ways until the S-box has the proper size.

27 Claims, 8 Drawing Sheets



the target S-box, the process of generating a new column is repeated. (Step 170).

Once the temporary S-box reaches the desired size, the process may terminate but other measures may be taken to decrease the exploitability of the S-box by cryptanalytic attack further. A determination is made whether the columns exhibit ideal distribution (Step 175). As mentioned earlier, columns have ideal weight distribution if they have Hamming weight approximately equal to half their length. Columns generated using bent functions, however, will have Hamming weight either equal to $2^{n-1}-2^{n/2-1}$ or $2^{n-1}+2^{n/2-1}$. If the columns do not exhibit ideal distribution, complementing all of the bits of a randomly selected column may improve the overall distribution (Step 180).

Once the columns of the S-box are evenly distributed, the process tests the distribution of the rows of the S-box. If the Hamming weight of the rows is not between a and b, affine functions are added to selected columns. (Step 187) The nonlinearity and correlation between columns of an S-box is unaffected by the addition of linear functions to the columns; however, the distribution of the rows may be improved by this technique.

If the rows of the S-box are within acceptable parameters, the XORs of pairs of rows are also tested in this manner (Step 190). If the Hamming weight of the XORs of pairs of rows are not between a and b, affine functions are again added to selected columns (Step 187).

In the process described above, columns are simply discarded if they do not have the desired minimum nonlinearity with respect to any combination of columns already in the S-box. An alternative to this method is the construction method described below and depicted in FIG. **5**.

In FIG. **5**, the construction process begins by generating an S-box having m inputs and k inputs, wherein k is somewhat smaller than n, the number of outputs of the desired S-box (Step 505). The S-box may be generated in any manner, however, as in the first embodiment, the initial S-box should be chosen to exhibit the desired operational characteristics. Temporary variables, ncols and z, are initialized (Step 510). Variable ncols will denote the number of columns in the current S-box. The variable z will indicate the number of candidate columns temporarily set aside.

The iterative process begins by generating a candidate column, f_z (Step 515). Column f_z is tested first using Test 1 (Step 525). In the example shown in FIG. **5**, Test 1 is performed by computing the minimum nonlinearity of all combinations of k−1 columns of the S-box. Any subset of k−1 columns may be used in Test 1, but for this example and ease of notation, it is assumed that the kth column is the column not used in the calculation. Column f_z "passes" Test 1 if the minimum nonlinearity of all of the combinations exceeds a minimum threshold. For example, if, as in the example described above, the target S-box will have 8 inputs and 32 outputs, the minimum threshold for nonlinearity may be set to 74.

If column f_z fails Test 1, the column is discarded and the process generates a new column (Step 515). If column f_z passes Test 1, however, the column is tested using Test 2 (Step 530). In Test 2, the candidate column is tested for nonlinearity with respect to all combinations of columns involving the column not used in Test 1 which, in this example, is column k. Column f_z "passes" Test 2 if the minimum nonlinearity of all of the combinations exceeds a minimum threshold.

If column f_z passes Test 2, the column is appended to the S-box, z is set to 1 (Step 562), and variable ncols is incremented by 1 (Step 560). The process determines whether the number of columns in the current S-box is equal to the target size (Step 570). If so, the process ends (Step 575). If not, the iterative process begins again by generating a new column f_z (Step 515).

If column f_z fails Test 2, the candidate column is stored rather than discarded (Step 532). A determination is made whether f_z is the first stored column (Step 535). If f_z is the first stored column, variable z is incremented by 1 (Step 540) and the process continues by generating a new column (Step 515).

If f_z is the second column to pass Test 1 but fail Test 2, the first and second columns are XORed together (Step 545). The result, $f_z \oplus f_{z-1}$ is tested for nonlinearity using Test 1 (Step 550). Verifying that $f_z \oplus f_{z-1}$ passes Test 1 involves only half the computation effort required to verify that a new candidate column passes both Tests 1 and 2 and, therefore, will typically decrease the time required to find columns for large S-boxes. If $f_z \oplus f_{z-1}$ passes Test 1, f_z replaces the kth column of the current S-box (Step 555). The variable z is reset to 1 (Step 565). The column $f_z = f_1$, is appended to the S-box and the number of columns, ncols, is incremented by 1 (Step 570). If $f_z \oplus f_{z-1}$ fails Test 1, the variable z is reset to 1 (Step 552) and the process generates a new candidate column.

This method could also be extended in any one of several different ways. For example, the process could save three or more columns that pass Test 1 but fail Test 2. Also, the process may be modified so that any two or more of the Test 2 failures may be XOR'd together to produce a new candidate column.

FIG. **6** illustrates a system consistent with the present invention. As shown in FIG. **6**, the present invention uses a processor **610** connected to one or more input/output (I/O) devices (**615** and **620**) via data links **602** and **604**. In general, I/O devices **615** and **620** can be any devices that are capable of passing information to or receiving data from processor **610**. By way of example only, I/O devices **615** and **620** may be monitors, keyboards, modems, printers, display devices or workstations. Each workstation can be a personal computer (PC) or other hardware that includes a visual display device and data entry device such as a keyboard or mouse. It should further be understood that FIG. **6** describes an exemplary network where each of the hardware components may be implemented by conventional, commercially available computer systems.

It will be apparent to those skilled in the art that various modifications and variations can be made in the methods and systems of the present invention without departing from the spirit or scope of the invention. For example, in addition to the tests for nonlinearity and correlation between columns disclosed other tests may be performed that are functionally equivalent. The true scope of the claims is defined by the following claims.

What is claimed is:

1. A method of ciphering digital data through use of a substitution box (S-box) capable of receiving m inputs and producing n outputs, where m and n are positive integers greater than 1, comprising the steps of:

(a) generating the S-box by:

(i) selecting desired operational characteristics for the S-box;

(ii) generating a first S-box having m inputs and having k outputs, wherein k is less than n and greater than 1;

(iii) generating a column corresponding to a Boolean function of m inputs;

(iv) determining the operational characteristics of a temporary S-box formed by the first S-box and the column;

(v) modifying the first S-box by appending the column, if the temporary S-box possesses the desired operational characteristics;

(vi) repeating steps (iii) through (v) until the first S-box has n outputs; and subsequently

(b) ciphering digital data using the first S-box having the n outputs.

2. A method according to claim 1, further including the steps of:

selecting a desired level nonlinearity, NL, for the S-box;

determining the nonlinearity of a temporary S-box formed by the first S-box and the column; and

modifying the first S-box by appending the column, if the nonlinearity of the temporary S-box is equal to or greater than NL.

3. A method according to claim 1, further including the steps of:

selecting a desired level of correlation between columns, C, for the S-box;

determining the correlation between columns of the temporary S-box formed by the first S-box and the column; and

modifying the first S-box by appending the column, if the correlation between columns of the temporary S-box is equal to or less than C.

4. A method according to claim 3, further including the steps of:

selecting a desired level nonlinearity, NL, for the S-box;

determining the nonlinearity of a temporary S-box formed by the first S-box and the column; and

modifying the first S-box by appending the column, if the nonlinearity of the temporary S-box is equal to or greater than NL.

5. A method according to claim 2, wherein the step of determining the nonlinearity further comprises the steps of:

determining the nonlinearity for each combination of columns of the temporary S-box; and

setting the nonlinearity of the temporary S-box equal to the minimum nonlinearity of the combinations.

6. A method according to claim 3, wherein the step of determining the level of correlation between columns further comprises the steps of:

determining the correlation for each combination of columns of the temporary S-box; and,

setting the level of correlation between columns of the temporary S-box equal to the maximum correlation of the set of combinations.

7. A method according to claim 1, further comprising the step of:

determining the average Hamming weight of the columns of the m×n S-box; and

complementing all the bits of randomly selected columns until the columns of the S-box have average Hamming weight approximately 2^{n-1}.

8. A method according to claim 1, wherein the step of generating a first S-box includes the step of

generating the first S-box using bent functions.

9. A method according to claim 1, wherein the step of generating a column includes the step of

generating the column using bent functions.

10. A method according to claim 9, wherein the step of generating a first S-box includes the step of

generating the first S-box using bent functions.

11. A method according to claim 10, further comprising the step of:

complementing all the bits of randomly selected columns of the S-box until half the columns of the S-box have Hamming weight equal to $2^{n-1}-2^{n/2-1}$ and half the columns have Hamming weight equal to $2^{n-1}+2^{n/2-1}$.

12. A method according to claim 1, further comprising the steps of:

selecting a minimum Hamming weight, a, for the rows of the S-box;

selecting a maximum Hamming weight, b, for the rows of the S-box; and

adding affine functions to the columns of the S-box until the weight of each row of the S-box is between a and b inclusive.

13. A method according to claim 12, further comprising the step of:

adding affine functions to the columns of the S-box until the weight of each XOR of pairs of rows is between a and b inclusive.

14. A method according to claim 1, further comprising the steps of:

(a) selecting a minimum Hamming weight, a, for the rows of the S-box;

(b) selecting a maximum Hamming weight, b, for the rows of the S-box;

(c) adding affine functions to the columns of the S-box until the weight of each row of the S-box is between a and b inclusive;

(d) adding affine functions to the columns of the S-box until the weight of each XOR of pairs of rows is between a and b inclusive; and

(e) repeating steps (c) through (d) until the weight of each row of the S-box and the weight of each XOR of pairs of rows of the S-box are between a and b inclusive.

15. A method according to claim 1, wherein the step of generating a first S-box further includes the step of

generating a column using the bent function, d(x), such that $d:\{0, 1\}^m \rightarrow \{0, 1\}$

and

$$D(w_7 \ldots w_0) = \begin{cases} A(w_5 \ldots w_0), & w_6 = 0,\ w_7 = 0 \\ B(w_5 \ldots w_0), & w_6 = 0,\ w_7 = 1 \\ C(w_5 \ldots w_0), & w_6 = 1,\ w_7 = 0 \\ -2^{2n}[A(w_5 \ldots W_0)B(w_5 \ldots w_0)C(w_5 \ldots w_0)]^{-1}, & w_6 = 1,\ w_7 = 1 \end{cases}$$

represents the Walsh transform of the function where a, b, c ϵB_6 and A, B, C are their respective Walsh Transforms.

16. A method according to claim 15, wherein the step of generating a column further includes the step of generating a column using the bent function, f(x), where f is the mapping $f:\{0, 1\}^m \rightarrow \{0, 1\}$ and f(x) may be further defined as:

$$f(x) = \frac{1}{2}\left(1 + \frac{1}{2^{n/2}} \sum_{w \in (0,1)^m} (-1)^{w \cdot x \oplus d(x)}\right).$$

17. A method according to claim 2, wherein the step of determining nonlinearity of an S-box includes the step of

determining the nonlinearity as:

$$nl(S) = \min_{f \in C} nl(f)$$

where C is the set of all linear combinations of the columns of S and

$$nl(f) = 2^{n-1} - \frac{1}{2} \max_{w \in (0,1)^m} |W(f)(w)|.$$

18. A method of ciphering digital data through use of an mxn substitution box (S-box), where m and n are positive integers greater than 1, comprising the steps of:

(a) generating the S-box by:

(i) selecting a desired level nonlinearity, NL, for the mxn S-box;

(ii) generating a fist S-box having m inputs and k outputs, wherein k is less than n and greater than 1;

(iii) generating a first column, C_1, of length m;

(iv) performing a first operational test using column C_1;

(v) performing a second operational test using column C_1;

(vi) generating a second column, C_2, of length m;

(vii) performing the first operational test using column C_2;

(viii) performing the second operational test using column C_2;

(ix) determining X_1, the XOR of C_1 and C_2 if C_1 and C_2 both pass the first operational test and fail the second operational test;

(x) performing the first operational test using X_1;

(xi) replacing the kth column of the first S-box with C_1 and appending C_2 to the first S-box as the $k+1^{st}$ column, i X_1 passes the first operational test;

(xii) incrementing k by 1;

(xiii) repeating steps (iii) through (xii) until k equals n; and

(b) ciphering digital data using an S-box having m inputs and k outputs where k=n.

19. A method according to claim 18, wherein the step of performing a first operational test further includes the step of:

determining the minimum nonlinearity of the set containing all combinations of k-1 columns of the first S-box and the column.

20. A method according to claim 18, wherein the step of performing a second operational test further includes the step of:

determining the minimum nonlinearity of the set containing all combinations of the columns of the first S-box that include the column omitted in the first operational test and the column.

21. A system for ciphering digital data using a substitution box (S-box) capable of receiving m inputs and producing n outputs, where m and n are positive integers greater than 1, comprising:

means for generating the S-box including:

means for selecting desired operational characteristics;

means for generating a first S-box having m inputs and having k outputs, wherein k is less than n and greater than 1;

means for generating a column corresponding to a Boolean function of m inputs;

means for determining the operational characteristics of a temporary S-box formed by the first S-box and the column;

means for modifying the first S-box by appending the column, if the temporary S-box possesses the desired operational characteristics; and

means for ciphering digital data using a modified first S-box.

22. A system for ciphering digital data using a substitution box (S-box) capable of receiving m inputs and producing n outputs, where m and n are positive integers greater than 1, comprising:

means for generating the S-box including:

means for selecting a desired level nonlinearity, NL, for the X-box;

means for generating a first S-box having nonlinearity equal to or greater than NL;

means for generating a column corresponding to a Boolean function of m inputs;

means for determining the nonlinearity of a temporary S-box formed by the first S-box and the column;

means for modifying the first S-box by appending the column, if the nonlinearity of the temporary S-box is equal to or greater than NL; and

means for ciphering digital data using a modified first S-box.

23. A system according to claim 22, further comprising:

means for selecting a desired level of correlation between columns, C, for the S-box;

means for determining the correlation between columns of the temporary S-box formed by the first S-box and the column; and

means for modifying the first S-box by appending the column, if the correlation between columns of the temporary S-box is equal to or less than C.

24. A system according to claim 22, further comprising:

means for determining the average Hamming weight of the columns of the mxn S-box; and

means for complementing all the bits of randomly selected columns until the columns of the S-box have average Hamming weight approximately 2^{n-1}.

25. A system for ciphering digital data using an mxn substitution box (S-box), where m and n are positive integers greater than 1, comprising:

means for generating the S-box including:

means for selecting a desired level nonlinearity, NL, for the mxn S-box;

means for generating a first S-box having m inputs and k outputs, wherein k is less than n and greater than 1;

means for generating columns, C_1 and C_2;

means for performing a first operational test;

means for performing a second operational test;

means for determining the XOR X_1 of the two columns C_1 and C_2;

means for replacing the kth column of the first S-box with C_1 and appending C_2 to the first S-box as the k+1 column, if X_1 passes the first operational test and if both C_1 and C_2 and pass the first operational test and fail the second operational test; and

means for ciphering digital data using an S-box resultant from the means for replacing.

26. A system according to claim 25, wherein the step of performing a first operational test further includes the step of:

determining the minimum nonlinearity of the set containing all combinations of k-1 columns of the first S-box and the column.

27. A system according to claim 24, wherein the step of performing a second operational test further includes the step of:

determining the minimum nonlinearity of the set containing all combinations of the columns of the first S-box that include the column omitted in the first operational test and the column.

* * * * *

28/3,K/46 (Item 46 from file: 350)
DIALOG(R)File 350:Derwent WPIX

011981544 **Image available**
WPI Acc No: 1998-398454/199834
Related WPI Acc No: 1998-087283
XRPX Acc No: N98-310011
Block cipher system for generation of variable S - boxes **from arbitrary keys of arbitrary length - utilises** initial **set of linearly independent numbers to generate** substitution **tables**
Patent Assignee: TELEDYNE IND INC (TDCO)
Inventor: GARCKEN K T; KISYLIA A P; STRAWBRIDGE C E
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 5778074	A	19980707	US 95676	A	19950629	199834 B
			US 96673437	A	19960628	

Priority Applications (No Type Date): US 95676 P 19950629; US 96673437 A 19960628
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5778074	A		18	H04L-009/06	Provisional application US 95676

Block cipher system for generation of variable S - boxes **from arbitrary keys of arbitrary length**...

...**utilises** initial **set of linearly independent numbers to generate** substitution **tables**

...Abstract (Basic): from a number of complete sets of linearly independent numbers.The system further includes an **encryption / decryption** key(11). A generation unit(13,15) is included to generate **portions** of a n-bit **encryption** table (E) and n-bit **decryption** table (D) from a fixed n-bit source **substitution** table (R) stored in memory.The unit also generates a complete set of linearly independent...

...USE- **Encrypts / decrypts** plaintext/ciphertext and used in digital **encryption** communication system such as in set-top boxes for the delivery of digital T.V...

...ADVANTAGE- Enhanced resistance to **cryptanalysis** .Allows rapid key changes...
...Title Terms: **INITIAL** ;
International Patent Class (Main): **H04L-009/06**
Manual Codes (EPI/S-X): **W01-A05A**

US005778074A

United States Patent [19]

Garcken et al.

[11] Patent Number: 5,778,074

[45] Date of Patent: Jul. 7, 1998

[54] **METHODS FOR GENERATING VARIABLE S-BOXES FROM ARBITRARY KEYS OF ARBITRARY LENGTH INCLUDING METHODS WHICH ALLOW RAPID KEY CHANGES**

[75] Inventors: **Knute T. Garcken**, Ventura; **Charles E. Strawbridge**, Camarillo; **Andrew Philip Kisylia**, Agoura Hills, all of Calif.

[73] Assignee: **Teledyne Industries, Inc.**, Newbury Park, Calif.

[21] Appl. No.: **673,437**

[22] Filed: **Jun. 28, 1996**

Related U.S. Application Data

[60] Provisional application No. 60/000,676, Jun. 29, 1995.

[51] **Int. Cl.**[6] **H04L 9/06**

[52] **U.S. Cl.** **380/37**; 380/28; 380/42

[58] **Field of Search** 380/21, 37, 42, 380/28, 29

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,195,196	3/1980	Feistel .	
4,431,865	2/1984	Bernede et al. .	
4,531,020	7/1985	Wechselberger et al. .	
4,751,733	6/1988	Delayaye et al. .	
4,803,725	2/1989	Horne et al. .	
4,897,876	1/1990	Davies .	
5,003,596	3/1991	Wood	380/28
5,214,704	5/1993	Mittenthal	380/37
5,237,611	8/1993	Rasmussen et al. .	
5,365,589	11/1994	Gutowitz .	
5,412,729	5/1995	Liu .	

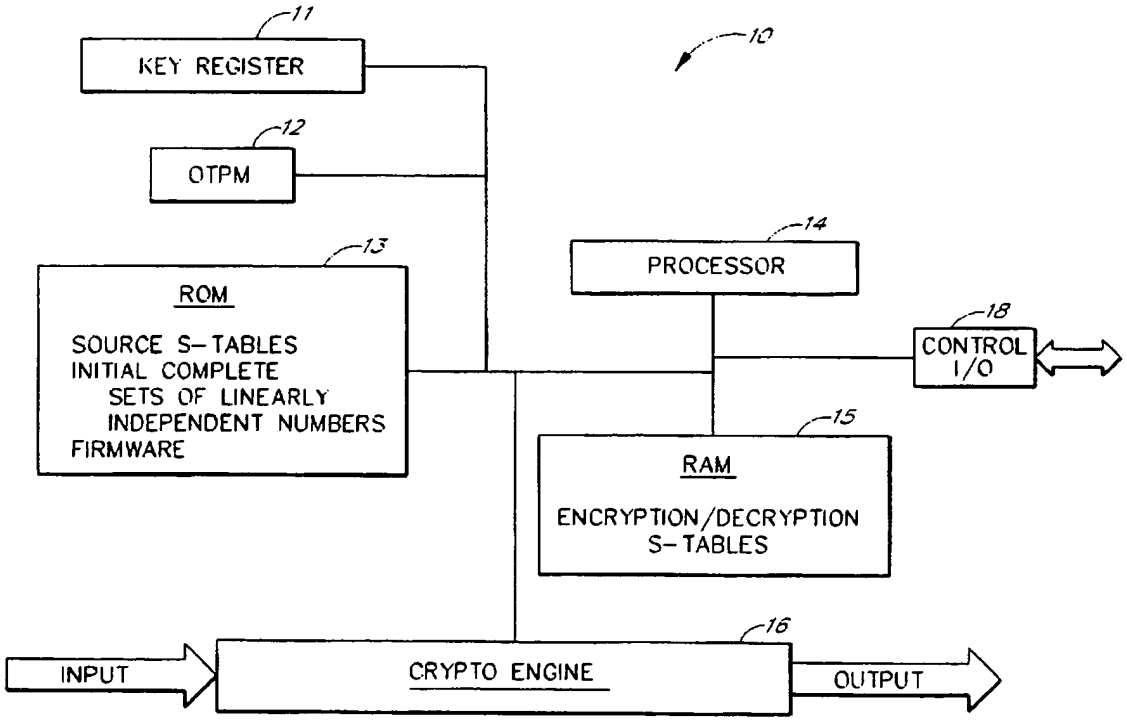
Primary Examiner—Gilberto Barrón, Jr.

Attorney, Agent, or Firm—Knobbe, Martens Olson & Bear, LLP

[57] **ABSTRACT**

A system for generating variable substitution boxes from arbitrary keys for use in a block cipher system utilizes an initial set of linearly independent numbers to generate substitution tables. The initial set of linearly independent numbers is modulated with the bits of an arbitrary key through operations that result in final sets of linearly independent numbers to form the substitution tables. The system also includes an implementation which allows for rapid key changes for the crypto system by only generating portions of the substitution tables as needed for specific blocks of input data to be encrypted or decrypted.

23 Claims, 7 Drawing Sheets



receive, and optionally send, high-speed digital data through television cable (CATV) networks which are capable of delivering digital data. As depicted in FIG. 4, the use of the present invention in a cable modem **400** connected to cable system **402** involves a duplex filter **404**, a tuner **406**, a Quadrature Phase Shift Keying (QPSK) modulator **408** a Quadrature Amplitude Modulation (QAM) demodulator **410**, a block cipher security system **412** complying with the present invention, a processor **414**, and a network interface such as an Ethernet interface **416** coupled to a computer **417**, and optionally a conventional telephone line modem **418** connected to the telephone lines **420**.

The cable modem system **400** receives data frames from the downstream RF channel **403** from the cable system **402**. The received frames, after qualification and processing, are delivered to the computer **417** via the network interface **416**. In the depicted embodiment, the network interface advantageously comprises a 10Base-T ethernet interface. Data received from the computer **417** ("client") (through the interface **416**) is formatted and returned upstream via the upstream modulator **408**. The modem can return data received from the client **417** via the optional modem **418**. This option provides the user tee benefit of hi-speed downstream data delivery when the user is using a "One Way" cable plant (i.e., there is no upstream ability in the cable system **402**).

For data from the Cable System **402**, the RF signal arrives at the duplex filter **404** which provides high-pass filtering. The signal is then delivered to the tuner **406**. The tuner selects the RF channel of interest and delivers the selected intermediate frequency (IF) signal to the QAM demodulator **410**. The QAM demodulator **410** demodulates the IF signal, providing synchronization, error detection/correction and outputs parallel data to the receiver portion (the receive buffer Rx) of the Security Device **412**. The security device **412** decrypts the received data, if necessary, and based on conditional access functionality contained in the security device **412**, and conditional access control information received in the downstream data, delivers the decrypted data to the processor **414**. The processor **414** is responsible for reassembling the received packets of data and, after additional qualification, signals the ethernet controller to send the packet(s) to the computer **417**.

For data to be sent upstream, the processor **414** formats the data received from the computer **417** for transmission via the QPSK modulator **408** or via the optional modem **418**. The processor then passes the data packet(s) to the security device **412** for encryption. The security device **412** then passes the packets to the QPSK modulator **408**, to the duplex filter **404**, and then to the cable system **402**. If the packet is to be sent via the standard modem **418**, the data packet is passed by the processor **414** to the modem **418** without encrypting.

The encryption/decryption functionality of the security device **412** may be implemented in software or in hardware. In the present embodiment, software can be used for data throughput requirements of less than 10 Mbits/sec. Hardware provides faster throughput. In order to handle packets from different sources, the security device **412** may be required to perform fast key switching. In applications where only a few simultaneous sources are possible, this may be accomplished by caching the tables required for each key in memory (such as RAM. In applications where numerous simultaneous sources are possible, or where the use of memory (such as RAM) is constrained, the embodiment of FIG. **3** above may be utilized.

While preferred embodiments of this invention have been disclosed herein, those skilled in the art will appreciate that changes and modifications may be made therein without departing from the spirit and scope of the invention.

We claim:

1. A block cipher system, in which sub-blocks of data are replaced by other sub-blocks as defined by one or more mappings, wherein each snapping can be expressed as a substitution table, said system comprising:

a first complete set of linearly independent numbers selected from a plurality of complete sets of linearly independent numbers;

a key; and

means for generating at least portions of a resulting n-bit encryption table (E) and a resulting n-bit decryption table (D) from a fixed n-bit source substitution table (R) stored in memory and said first complete set of linearly independent n-bit numbers.

2. The block cipher system of claim **1**, wherein said first complete set of linearly independent n-bit numbers is used to form a linear transformation for the source substitution table (R).

3. The block cipher system of claim **2**, wherein the linear transformation comprises a second complete set of linearly independent numbers generated by modulating the first complete set of linearly independent numbers with said key.

4. The block cipher system of claim **3**, wherein said linear transformation (T) is used as follows:

For K from 0 through 2^n-1:

$$E[T(K)]=T(R[K]),$$

and

$$D[T(R[K])]=T(K).$$

5. The block cipher system of claim **4**, wherein the transformation (T) comprises a right multiplication by a matrix formed from the second complete act of linearly independent numbers.

6. The block cipher system of claim **3**, wherein said linear transformation is used as follows:

For K from 0 through 2^n-1:

$$E[T(K)]=T(R[K]\oplus F),$$

and

$$D[T(R[K]\oplus F)]=T(K),$$

where F is an n-bit value determined from the key.

7. The block cipher system of claim **1**, wherein said means for generating comprises a means for performing a linear transformation (T) on said source substitution table (R), said transformation (T) comprising a second complete set of linearly independent numbers generated from said key and said first complete set of linearly independent numbers.

8. The block cipher system of claim **1**, wherein said means for generating comprises means for concurrently generating a second complete set of linearly independent n-bit numbers to form a first linear transformation (T), and a third complete set of linearly independent n-bit numbers to form a second linear transformation (T^{-1}) which is the inverse of the first linear transformation (T).

9. The block cipher system of claim **1**, wherein said means for generating comprises means for generating the specific n-bit output which correspond to outputs for the encryption substitution table or the decryption substitution table on an as needed basis, for each n-bit input value (U) without generating the entire encryption substitution table (E) or entire substitution table (D).

10. The block cipher system of claim 9, wherein said means for generating further comprises encryption and decryption source substitution tables, R_E and R_D, stored in memory, and further comprises means for performing an n-bit transformation, T, and its inverse, T^{-1}, as follows:

$$E[U]=T(R_E[T^{-1}(U)]),$$

and

$$D[U]=T(R_D[T^{-1}(U)]).$$

11. A block cipher system comprising:

a first complete set of linearly independent numbers, each of a selected bit length;

a key;

a source substitution table stored in memory;

a modulation module responsive to selected bits from said key to control operations on said first complete set of linearly independent numbers to obtain a second complete set of linearly independent numbers;

a transformation module which transforms the source substitution table stored in memory using said second complete set of linearly independent numbers to obtain a resulting substitution table; and

a decryption substitution module which has an input and an output, said input comprising data blocks for which substitution is desired and said output comprising the substitution blocks for said input data blocks, said substitution blocks obtained from said resulting substitution table.

12. The block cipher system of claim 11, further comprising an encryption module with an input and an output, said input comprising data blocks for which substitution is desired and said output comprising substitution blocks for said input data blocks, said substitution blocks obtained from said resulting substitution table.

13. The block cipher system of claim 11, further comprising a plurality of complete sets of linearly independent numbers stored in memory, wherein said modulation module is responsive to selected key bits to select said first complete set of linearly independent numbers from said plurality of complete sets.

14. The block cipher system of claim 13, wherein said modulation module is responsive to other key bits to select certain numbers from said first complete set of linearly independent numbers for XOR operations with other numbers from said first complete set of linearly independent numbers to form said second set of linearly independent numbers.

15. The block cipher system of claim 11, wherein said modulation module is responsive to selected key bits to select certain numbers of said first set of linearly independent numbers for an XOR operation with other numbers of said first set of linearly independent numbers to form said second set of linearly independent numbers.

16. The block cipher system of claim 11, wherein said transformation module forms a matrix of the second set of linearly independent numbers and uses this matrix as a transformation of the source substitution table to form said resulting substitution table.

17. The block cipher system of claim 16, wherein said transformation module right multiplies data from said source substitution table by said matrix to form said resulting substitution table.

18. The block cipher system of claim 17, wherein said source substitution table comprises a plurality of data blocks of a predetermined bit length, and wherein said transformation comprises two inputs, said first input being an index input and said second input being said data blocks from said source substitution table, wherein said transformation module right multiplies said index by said matrix and right multiplies said data blocks by said matrix in order to obtain said resulting substitution table.

19. The block cipher system of claim 18, wherein said transformation module comprises outputs, said outputs comprising a transformed index and a transformed data block, said index providing an address for the transformed data block.

20. A block cipher system comprising:

a first complete set of linearly independent numbers, each of a selected bit length;

a key;

a source substitution table stored in memory;

a transformation module which transforms the source substitution table stored in memory using a transformation from said first complete set of linearly independent numbers and said key to obtain a temporary portions of a resulting substitution table on an as needed basis, without generating entire substitution tables for encryption and decryption; and

a crypto module which has an input and an output, sad input comprising data blocks to be encrypted or decrypted and said output comprising substitution blocks for said input data blocks, said substitution blocks obtained from said temporary portions of the resulting substitution table.

21. The block cipher system of claim 20, wherein said transformation module comprises an n-bit transformation logic and an n-bit inverse transformation logic.

22. The block cipher system of claim 21, wherein said n-bit linear transformation logic and said n-bit inverse linear transformation logic have variable portions which are configured simultaneously.

23. The block cipher system of claim 21, wherein said transformation module performs the following transformation:

$$E[U]=T(R_E[T^{-1}(U)]),$$

and

$$D[U]=T(R_D[T^{-1}(U)]),$$

where R_E is the source encryption substitution table, R_D is the source decryption substitution table, T^{-1} is the inverse n-bit linear transformation, T is the n-bit linear transformation, E[U] is the temporary portion of the resulting encryption substitution table and D[U] is the temporary portion of the resulting decryption table and U is the input data block.

* * * * *

28/3,K/31 (Item 31 from file: 350)
DIALOG(R)File 350:Derwent WPIX

013990571 **Image available**
WPI Acc No: 2001-474786/200151
XRPX Acc No: N01-351382
Computer-readable code for providing a byte symmetric key block cipher, has computer-readable program code section **used for treating** substituted **bytes as input data bytes for subsequent iteration**
Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: COPPERSMITH D; GENNARO R; HALEVI S; JUTLA C S; MATYAS S M; PEYRAVIAN M; SAFFORD D R; ZUNIC N
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 6192129	B1	20010220	US 9818630	A	19980204	200151	B

Priority Applications (No Type Date): US 9818630 A 19980204
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6192129	B1		26	H04L-009/06	

... **readable code for providing a byte symmetric key block cipher, has computer-readable program code** section **used for treating** substituted **bytes as input data bytes for subsequent iteration**

Abstract (Basic):
... A byte value is determined by performing a third XOR operation followed by a **second S - box** lookup operation to create multiple **substituted** bytes. A computer-readable program code **section** is used for treating the **substituted** bytes as input data bytes for a subsequent iteration of the program code **section** .
... For providing a byte symmetric key block cipher for **encryption** and **decryption** in computer system...

...Improves **encryption** strength while enhancing **encryption** efficiency. Maximizes the number of environments in which solution can be used. Enables efficient and error-free **decryption** of **encrypted** data...

...The figure shows the flowchart of a logic used for data block **encryption** .

...Title Terms: **SECTION** ;
International Patent Class (Main): **H04L-009/06**
Manual Codes (EPI/S-X): **T01-D01** ...

... **T01-S01C** ...
... **T01-S03** ...

... **W01-A05A** ...

... **W01-A06B5A** ...

... **W01-A06G3**

US006192129B1

(12) **United States Patent**
Coppersmith et al.

(10) **Patent No.: US 6,192,129 B1**
(45) **Date of Patent: *Feb. 20, 2001**

(54) **METHOD AND APPARATUS FOR ADVANCED BYTE-ORIENTED SYMMETRIC KEY BLOCK CIPHER WITH VARIABLE LENGTH KEY AND BLOCK**

(75) Inventors: **Don Coppersmith**, Ossining; **Rosario Gennaro**, New York; **Shai Halevi**, Heartsdale; **Charanjit S. Jutla**, Elmsford, all of NY (US); **Stephen M. Matyas, Jr.**, Manassas, VA (US); **Mohammed Peyravian**, Cary, NC (US); **David Robert Safford**, Brewster, NY (US); **Nevenko Zunic**, Wappingers Falls, NY (US)

(73) Assignee: **International Business Machines Corporation**, Armonk, NY (US)

(*) Notice: Under 35 U.S.C. 154(b), the term of this patent shall be extended for 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/018,630**

(22) Filed: **Feb. 4, 1998**

(51) **Int. Cl.**[7] **H04L 9/06**
(52) **U.S. Cl.** **380/259**; 380/37; 380/29
(58) **Field of Search** 380/29, 37, 259, 380/264, 265

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,375,579		3/1983	Davida et al. .	
5,432,848		7/1995	Butter et al. .	
5,481,613		1/1996	Ford et al. .	
5,513,262		4/1996	van Rumpt et al. .	
5,548,648		8/1996	Yorke-Smith .	
5,623,549	*	4/1997	Ritter	380/37
5,666,414		9/1997	Micali .	
5,673,319		9/1997	Bellare et al. .	
5,825,886	*	10/1998	Adams et al.	380/37 X

OTHER PUBLICATIONS

"Description of a New Variable–Length Key, 64–Bit Block Cipher (Blowfish)", Bruce Schneier Counterpane Systems, Oak Park, IL, http://www.cryptocard.com/algorith/blowfish.html/Internet Article Retrieved Jan. 22, 1998 dated Jan. 30, 1996.

Dr. Dobb's Journal, Apr. 1994, vol. 20, issue 4, "The Blowfish Encryption Algorithm", pp. 38, 40, 98–99.

Cambridge Security Workshop, Cambridge, U.K., Dec. 9–11, 1993 Proceedings, "Safer K–64: A Byte–Oriented Block–Ciphering Algorithm", J.L. Massey, pp. 1–17.

Second International Workshop, Leuven, Belgium, Dec. 14–16, 1994 Proceedngs, "The RC5 Encryption Algorithm", R. L. Rivest, pp 86–96.

15th Annual International Cryptology Conference, Santa Barbara, CA, Aug. 27–31, 1995, "On Differential and Linear Cryptanalysis of the RC5 Encryption Algorithm", B.S. Kalinski and Y.L. Yin, pp 171–184.

5th IMA Conference, Cirencester, UK, Dec. 18–20, 1995 Proceedings, "A Broadcast Key Distribution Scheme Based on Block Designs", V. Korjik, M. Ivkov, Y. Merinovich, A. Barg, H, van Tilborg, pp 3–12.

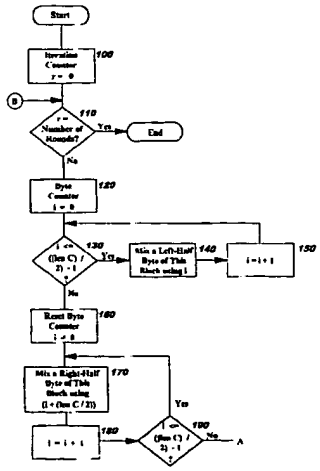
* cited by examiner

Primary Examiner—Gilberto Barrón, Jr.
(74) *Attorney, Agent, or Firm*—Jeanine S. Ray-Yarletts; Marcia L. Doubet

(57) **ABSTRACT**

A method and apparatus for an advanced byte-oriented symmetric key cipher for encryption and decryption, using a block cipher algorithm. Different block sizes and key sizes are supported, and a different sub-key is used in each round. Encryption is computed using a variable number of rounds of mixing, permutation, and key-dependent substitution. Decryption uses a variable number of rounds of key-dependent inverse substitution, inverse permutation, and inverse mixing. The variable length sub-keys are data-independent, and can be precomputed.

27 Claims, 10 Drawing Sheets



operand is located by effectively rotating the current sub-key one byte to the right, then using the byte from this rotated sub-key that corresponds to the byte counter i. The second operand is the byte of the current sub-key pointed to by the byte counter (without having rotated the sub-key). For example, if the sub-keys and blocks are 8 bytes long, and the byte counter is 0, the first operand will be the byte numbered "i−1 mod ‖C‖", which in this case evaluates to ((0−1) mod 8), or 7. This is the eighth byte of the current sub-key. The second operand will be the first byte, the byte numbered 0, of the same sub-key.

At Step **630**, the generation counter v is incremented. Control then transfers back to Step **580**.

Control reaches Step **640** when all the iterations of mixing, permutation, and key-dependent substitution have completed for this sub-key. At this step, the current sub-key is exclusive OR'd with the temporary variable X in which a value was saved at Step **560**. The result of the exclusive OR is substituted as the new value of the current sub-key.

At Step **650**, the byte counter i is again initialized to 0. Step **660** compares the byte counter value to the length of the blocks. If the test at Step **660** has a positive result, control transfers to Step **670**; otherwise, control transfers to Step **690**.

Step **670** takes a byte from the newly-generated sub-key, and substitutes it back into the original input key, which results in further randomization of the sub-keys being generated. The following mathematical equation defines the process by which this is done:

$$K_{rL+i \bmod \|K\|} = K_i^{<r>}$$

The byte counter i points to the current byte of the current sub-key, and the iteration counter r identifies the current sub-key. This byte from the sub-key will be substituted into the input key. The position at which this byte will be substituted is determined by the expression "rL+i mod ‖K‖". Using the same example used above for inserting the byte at Step **540**, where the block size C is 8 bytes, the input key K is 24 bytes long, the total number of rounds of processing is 11, the iteration counter r is 0, and the byte counter i is 0, the result is that the first byte of the first sub-key is substituted into the first byte of the input key. If the iteration counter r is 3, the byte counter i is 2, and the other variables are unchanged, the expression becomes ((3 * 2)+(2 mod 24))= (6+2)=8, so that the third byte (because i=2) of the fourth sub-key (because r=3 is substituted for the ninth byte (the byte numbered 8) of the input key.

At Step **680**, the byte counter i is incremented. Control then transfers back to Step **660**.

Control reaches Step **690** when a complete round of sub-key generation (consisting of generating the sub-key bytes, encrypting the bytes, and substituting the encrypted bytes back into the input key) has completed. At Step **690**, the iteration counter r is incremented. Control then transfers back to Step **510**.

While the preferred embodiments of the present invention have been described, additional variations and modifications in those embodiments may occur to those skilled in the art once they learn of the basic inventive concepts. Therefore, it is intended that the appended claims shall be construed to include both the preferred embodiments and all such variations and modifications as fall within the spirit and scope of the invention.

TABLE 1

SYMBOL	DEFINITION
C	The plaintext (input data) or ciphertext (encrypted) block.
‖C‖	The length of C in bytes, where ‖C‖ is an even integer and ‖C‖ ≧ 8.
C_i	Byte i of C, where $0 \leqq i \leqq \|C\| - 1$.
R	An integer number denoting the total number of rounds of the encryption algorithm, where $R \geqq \lceil (\|C\| + 16)/5 \rceil$. The notation $\lceil x \rceil$ denotes the smallest integer greater than or equal to x. For example, if $x = 3.2$, then $\lceil x \rceil = 4$.
K	The symmetric (secret) encryption/decryption input key.
‖K‖	The length of K in bytes. ‖K‖ is an integer in the following interval: $\|C\| \leqq \|K\| \leqq \|C\| \times R$.
K_j	Byte j of key K, where $0 \leqq j \leqq \|K\| - 1$.
$K^{<r>}$	The rth sub-key derived from K, where $0 \leqq r \leqq R - 1$. Each sub-key is of length ‖C‖ bytes.
$K_i^{<r>}$	Byte i of sub-key $K^{<r>}$, where $0 \leqq i \leqq \|C\| - 1$.
L	An integer defined as $L = \lceil (\|K\| - \|C\|)/(R - 1) \rceil$. When the input key is bigger than the block size, L equally divides the additional input key bytes among the sub-keys.
$S_j^{<n>}$	The jth entry of the nth s-box, where $0 \leqq n \leqq 1$, and $0 \leqq j \leqq 255$. Each s-box contains 256 non-repeating 8-bit values which are indexed from 0 to 255. The jth entry of the inverse of the nth s-box is denoted by $S_j^{-<n>}$.
A ⇔ B	"⇔" denotes swapping of A with B.
A ⊕ B	"⊕" denotes exclusive ORing A and B.

What is claimed is:

1. In a computing environment, computer-readable code for providing a byte-oriented symmetric key block cipher which supports a variable length symmetric input key, a variable length block, and a variable number of rounds, said computer-readable code embodied on a computer-readable medium and comprising:

computer-readable program code means for determining a number of rounds of cipher processing to use as said variable number of rounds, a key length of said variable length symmetric input key, and a block length of said variable length block;

computer-readable program code means for generating a plurality of sub-keys using said symmetric input key as an input value, wherein each of said generated sub-keys is equal in length to said block length and where a distinct one of said sub-keys is generated for each of said number of rounds;

computer-readable program code means for obtaining an input data block to be encrypted, wherein said input data block comprises a plurality of input data bytes, said plurality being equal in number to said block length; and

computer-readable program code means for iteratively performing a set of round functions a number of times equal to said number of rounds in order to encrypt said input data block, wherein said set of round functions comprises a mixing function, a permuting function, and a key-dependent substitution function, and wherein said computer-readable program code means for iteratively performing further comprises:

computer-readable program code means for performing said mixing function by mixing each of said input data bytes using a first XOR operation and a second XOR operation, wherein said first and second XOR operations are different, followed by a first substitution-box (S-box) lookup operation, thereby creating a plurality of mixed bytes;

computer-readable program code means for performing said permuting function by swapping each of said mixed bytes, thereby creating a plurality of permuted bytes;

computer-readable program code means for performing said key-dependent substitution function by substituting a byte value for each of said permuted bytes, wherein said byte value is determined by performing a third XOR operation followed by a second S-box lookup operation, thereby creating a plurality of substituted bytes; and

computer-readable program code means for treating said plurality of substituted bytes as said plurality of input data bytes for a subsequent iteration of said computer-readable program code means for iteratively performing, provided said number of times has not been reached.

2. The computer-readable code according to claim 1, wherein said computer-readable program code means for performing said mixing function further comprises:

computer-readable program code means for dividing said plurality of input data bytes into a left input half and a right input half;

computer-readable program code means for performing a first mixing operation on said left input half and a second mixing operation on said right input half, wherein said second mixing operation uses a different selection of operands for said first and second XOR operations than does said first mixing operation;

computer-readable program code means for using each byte of a result of said second XOR operation of said first mixing operation as a lookup index for said first S-box lookup operation to retrieve bytes of a new left half; and

computer-readable pro=am code means for using each byte of an output of said second XOR operation of said second mixing operation as said lookup index for said first S-box lookup operation to retrieve bytes of a new right half.

3. The computer-readable code according to claim 2, wherein:

said computer-readable program code means for performing said first mixing operation further comprises:

computer-readable program code means for using an identically-numbered byte from said left input half and said right input half as operands of said first XOR operation; and

computer-readable program code means for using a result of said first XOR operation and a byte from said right input half that has been effectively rotated right one byte as operands of said second XOR operation; and

said computer-readable program code means for performing said second mixing operation further comprises:

computer-readable program code means for using a selected byte from said right input half and a previously-mixed byte from said new left half that has been effectively rotated right one byte as operands of said first XOR operation; and

computer-readable program code means for using an output of said first XOR operation and a different previously-mixed byte from said new left half that has been effectively rotated left two bytes as operands of said second XOR operation.

4. The computer-readable code according to claim 1, wherein said computer-readable program code means for performing said mixing function and said computer-readable program code means for performing said key-dependent substitution function perform said first S-box lookup operation and said second S-box lookup operational, respectively, by accessing a selected one of two distinct S-boxes using a one-byte index, each of said S-boxes having 256 distinct entries, each of said entries being a one-byte value.

5. The computer-readable code according to claim 1, wherein one or more of said computer-readable program code means is embodied in a hardware chip.

6. The computer-readable code according to claim 1, wherein said computer-readable program code means for performing said permuting function further comprises:

computer-readable program code means for dividing said plurality of mixed bytes into a left mixed half and a right mixed half; and

computer-readable program code means for swapping said left mixed half with said right mixed half.

7. The computer-readable code according to claim 1, wherein said computer-readable program code means for performing said key-dependent substitution function further comprises:

computer-readable program code means for using a sub-key byte from a selected one of said generated sub-keys which is uniquely associated with said round as an operand of said third XOR operation, along with said each permuted byte; and

computer-readable program code means for performing said second S-box lookup operation using each byte of a result of said third XOR operation as an index.

8. The computer-readable code according to claim 1, wherein particular values of one or more of said number of rounds, said key length, and said block length are determined in advance in order to optimize said computer-readable code, and wherein said computer-readable program code means for determining therefore operates as if said one or more particular values are fixed.

9. The computer-readable code according to claim 1, further comprising:

computer-readable program code means for decrypting said encrypted data block, resulting in restoration of said plurality of input data bytes, by performing a set of inverse round functions said number of times equal to said number of rounds, wherein said set of inverse round functions comprises an inverse key-dependent substitution function which is inverse to said key-dependent substitution function, an inverse permuting function which is inverse to said permuting function, and an inverse mixing function which is inverse to said mixing function.

10. A system for providing a byte-oriented symmetric key block cipher which supports a variable length symmetric input key, a variable length block, and a variable number of rounds, comprising:

means for determining a number of rounds of cipher processing to use as said variable number of rounds, a key length of said variable length symmetric input key, and a block length of said variable length block;

means for generating a plurality of sub-keys using said symmetric input key as an input value, wherein each of said generated sub-keys is equal in length to said block length and where a distinct one of said sub-keys is generated for each of said number of rounds;

means for obtaining an input data block to be encrypted, wherein said input data block comprises a plurality of input data bytes, said plurality being equal in number to said block length; and

means for iteratively performing a set of round functions a number of times equal to said number of rounds in order to encrypt said input data block, wherein said set

of round functions comprises a mixing function, a permuting function, and a key-dependent substitution function, and wherein said means for iteratively performing further comprises:

means for performing said mixing function by mixing each of said input data bytes using a first XOR operation and a second XOR operation, wherein said first and second XOR operations are different, followed by a first substitution-box (S-box) lookup operation, thereby creating a plurality of mixed bytes;

means for performing said permuting function by swapping each of said mixed bytes, thereby creating a plurality of permuted bytes;

means for performing said key-dependent substitution function by substituting a byte value for each of said permuted bytes, wherein said byte value is determined by performing a third XOR operation followed by a second S-box lookup operation, thereby creating a plurality of substituted bytes; and

means for treating said plurality of substituted bytes as said plurality of input data bytes for a subsequent iteration of said means for iteratively performing, provided said number of times has not been reached.

11. The system according to claim **10**, wherein said means for performing said mixing function further comprises:

means for dividing said plurality of input data bytes into a left input half and a right input half;

means for performing a first mixing operation on said left input half and a second mix operation said if right half, wherein said second mixing operation uses a different selection of operands for said first and said second XOR operations than does said first mixing operation;

means for using each byte of a result of said second XOR operation of said first mixing operation as a lookup index for said first S-box lookup operation to retrieve bytes of a new left half; and

means for using each byte of an output of said second XOR operation of said second mixing operation as said lookup index for said first S-box lookup operation to retrieve bytes of a new right half.

12. The system according to claim **11**, wherein:

said means for performing said first mixing operation further comprises:

means for using an identically-numbered byte from said left input half and said right input half as operands of said first XOR operation; and

means for using a result of said first XOR operation and a byte from said right input half that has been effectively rotated right one byte as operands of said second XOR operation; and

said means for performing said second mixing operation further comprises:

means for using a selected byte from said right input half and a previously-mixed byte from said new left half that has been effectively rotated right one byte as operands of said first XOR operation; and

means for using an output of said first XOR operation and a different previously-mixed byte from said new left half that has been effectively rotated left two bytes as operands of said second XOR operation.

13. The system according to claim **10**, wherein said means for performing said mixing function and said means for performing said key-dependent substitution function perform said first S-box lookup operation and said second S-box lookup operation, respectively, by accessing a selected one of two distinct S-boxes using a one-byte index, each of said S-boxes having 256 distinct entries, each of said entries being a one-byte value.

14. The system according to claim **10**, wherein one or more of said means is embodied in a hardware chip.

15. The system according to claim **10**, wherein said means for performing said permuting function further comprises:

means for dividing said plurality of mixed bytes into a left mixed half and a right mixed half; and

means for swapping said left mixed half with said right mixed half.

16. The system according to claim **10**, wherein said means for performing said key-dependent substitution function further comprises:

means for using a sub-key byte from a selected one of said generated sub-keys which is uniquely associated with said round as an operand of said third XOR operation, along with said each permuted byte; and

means for performing said second S-box lookup operation using each byte of a result of said third XOR operation as an index.

17. The system according to claim **10**, wherein particular values of one or more of said number of rounds, said key length, and said block length are determined in advance in order to optimize said system, and wherein said means for determining therefore operates as if said one or more particular values are fixed.

18. The system according to claim **10**, further comprising:

means for decrypting said encrypted data block, resulting in restoration of said plurality of input data bytes, by performing a set of inverse round functions said number of times equal to said number of rounds, wherein said set of inverse round functions comprises an inverse key-dependent substitution function which is inverse to said key-dependent substitution function, an inverse permuting function which is inverse to said permuting function, and an inverse mixing function which is inverse to said mixing function.

19. A method of providing a byte-oriented symmetric key block cipher which supports a variable length symmetric input key, a variable length block, and a variable number of rounds, comprising the steps of:

determining a number of rounds of cipher processing to use as said variable number of rounds, a key length of said variable length symmetric input key, and a block length of said variable length block;

generating a plurality of sub-keys using said symmetric input key as an input value, wherein each of said generated sub-keys is equal in length to said block length and where a distinct one of said sub-keys is generated for each of said number of rounds;

obtaining an input data block to be encrypted, wherein said input data block comprises a plurality of input data bytes, said plurality being equal in number to said block length; and

iteratively performing a set of round functions a number of times equal to said number of rounds in order to encrypt said input data block, wherein said set of round functions comprises a mixing function, a permuting function, and a key-dependent substitution function, and wherein said iteratively performing step further comprises the steps of:

performing said mixing function by mixing each of said input data bytes using a first XOR operation and a second XOR operation, wherein said first and second XOR operations are different, followed by a first

substitution-box (S-box) lookup operation, thereby creating a plurality of mixed bytes;

performing said permuting function by swapping each of said mixed bytes, thereby creating a plurality of permuted bytes;

performing said key-dependent substitution function by substituting a byte value for each of said permuted bytes, wherein said byte value is determined by performing a third XOR operation followed by a second S-box lookup operation, thereby creating a plurality of substituted bytes; and

treating said plurality of substituted bytes as said plurality of input data bytes for a subsequent iteration of said iteratively performing step, provided said number of times has not been reached.

20. The method according to claim **19**, wherein said step of performing said mixing function further comprises the steps of:

dividing said plurality of input data bytes into a left input half and a right input half;

performing a first mixing operation on said left input half and a second mixing operation on said right half, wherein said second mixing operation uses a different selection of operands for said first and second XOR operations than does said first mixing operation;

using each byte of a result of said second XOR operation of said first mixing operation as a lookup index for said first S-box lookup operation to retrieve bytes of a new left half; and

using each byte of an output of said second XOR operation of said second mixing operation as said lookup index for said first S-box lookup operation to retrieve bytes of a new right half.

21. The method according to claim **20**, wherein:

said step of performing said first mixing operation further comprises the steps of:

using an identically-numbered byte from said left input half and said right input half as operands of said first XOR operation; and

using a result of said first XOR operation and a byte from said right input half that has been effectively rotated right one byte as operands of said second XOR operation; and

said step of performing said second mixing operation further comprises the steps of:

using a selected byte from said right input half and a previously-mixed byte from said new left half that has been effectively rotated right one byte as operands of said first XOR operation; and

using an output of said first XOR operation and a different previously-mixed byte from said new left half that has been effectively rotated left two bytes as operands of said second XOR operation.

22. The method according to claim **19**, wherein said step of performing said mixing function and said step of performing said key-dependent substitution function perform said first S-box lookup operation and said second S-box lookup operation, respectively, by accessing a selected one of two distinct S-boxes using a one-byte index, each of said S-boxes having 256 distinct entries, each of said entries being a one-byte value.

23. The method according to claim **19**, wherein one or more of said steps is embodied in a hardware chip.

24. The method according to claim **19**, wherein said step of performing said permuting function further comprises the steps of

dividing said plurality of mixed bytes into a left mixed half and a right mixed half; and

swapping said left mixed half with said right mixed half.

25. The method according to claim **19**, wherein said step of performing said key-dependent substitution function further comprises the steps of:

using a sub-key byte from a selected one of said generated sub-keys which is uniquely associated with said round as an-operand of said third XOR operation, along with said each permuted byte; and

performing said second S-box lookup operation using each byte of a result of said third XOR operation as an index.

26. The method according to claims **19**, wherein particular values of one or more of said number of rounds, said key length, and said block length are determined in advance in order to optimize said method, and wherein said step of determining therefore operates as if said one or more particular values are fixed.

27. The method according to claim **19**, further comprising the step of:

decrypting said encrypted data block, resulting in restoration of said plurality of input data bytes, by performing a set of inverse round functions said number of times equal to said number of rounds, wherein said set of inverse round functions comprises an inverse key-dependent substitution function which is inverse to said key-dependent substitution function, an inverse permuting function which is inverse to said permuting function, and an inverse mixing function which to said mixing function.

* * * * *

28/3,K/48 (Item 48 from file: 350)
DIALOG(R)File 350:Derwent WPIX

011395233 **Image available**
WPI Acc No: 1997-373140/199734
XRPX Acc No: N97-309821
Inter-round mixing in iterated block substitution **systems - using trickle and quick trickle permutations for inter round permutations of sub blocks or individual bits to obtain respectively row completeness or quasi row completeness in Latin squares**
Patent Assignee: TELEDYNE ELECTRONIC TECHNOLOGIES (TDCO); TELEDYNE IND INC (TDCO)
Inventor: MITTENTHAL L
Number of Countries: 075 Number of Patents: 006
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
WO 9725799	A1	19970717	WO 97US367	A	19970103	199734 B
AU 9721116	A	19970801	AU 9721116	A	19970103	199748
			WO 97US367	A	19970103	
TW 337628	A	19980801	TW 97102649	A	19970305	199849
US 5838796	A	19981117	US 96584523	A	19960111	199902
			US 97888884	A	19970707	
US 5838794	A	19981117	US 96584523	A	19960111	199902
US 5838795	A	19981117	US 96584523	A	19960111	199902
			US 97888454	A	19970707	

Priority Applications (No Type Date): US 96584523 A 19960111; US 97888454 A 19970707; US 97888884 A 19970707
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 9725799	A1	E	125	H04L-009/06	
Designated States (National): AL AM AT AU AZ BA BB BG BR BY CA CH CN CU CZ DE DK EE ES FI GB GE HU IL IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK TJ TM TR TT UA UG US UZ VN				
Designated States (Regional): AT BE CH DE DK EA ES FI FR GB GR IE IT KE LS LU MC MW NL OA PT SD SE SZ UG					
AU 9721116	A			H04L-009/06	Based on patent WO 9725799
US 5838796	A			H04L-009/28	Div ex application US 96584523
US 5838795	A			H04L-009/28	Div ex application US 96584523
TW 337628	A			H04L-009/00	
US 5838794	A			H04L-009/28	

Inter-round mixing in iterated block substitution **systems**...

...Abstract (Basic): The method of **encryption** involves receiving successive blocks of data, each being sub- **divided** into sub-blocks of data. Each sub-block is assigned to one of the individual **substitution boxes** . A statistically optimised permutation is selected...

...applied, otherwise an exponent of one to the permutation is applied. After each round of **encryption** , an output of each numbered **substitution box** is applied as an input to the **substitution box** whose number is indicated by the permutation. The last two stages are repeated for a...

...USE/ADVANTAGE - Iterated block **substitution** system in which block **substitution** tables and pattern of inter round mixing are changed frequently. Interactions between sub blocks enhance...
...Title Terms: **SUBSTITUTE** ;

International Patent Class (Main): **H04L-009/00** ...

... **H04L-009/06** ...

... **H04L-009/28**
Manual Codes (EPI/S-X): **W01-A05**

US005838796A

United States Patent [19]

Mittenthal

[11] Patent Number: 5,838,796

[45] Date of Patent: Nov. 17, 1998

[54] **STATISTICALLY OPTIMIZED BIT PERMUTATIONS IN INTERATED BLOCK SUBSTITUTION SYSTEMS**

[75] Inventor: **Lothrop Mittenthal**, Thousand Oaks, Calif.

[73] Assignee: **Teledyne Industries, Inc.**, Los Angeles, Calif.

[21] Appl. No.: **888,884**

[22] Filed: **Jul. 7, 1997**

Related U.S. Application Data

[62] Division of Ser. No. 584,523, Jan. 11, 1996.

[51] **Int. Cl.**[6] **H04L 9/28**

[52] **U.S. Cl.** **380/28**; 380/37; 380/42

[58] **Field of Search** 380/28, 29, 37, 380/42, 59

[56] **References Cited**

U.S. PATENT DOCUMENTS

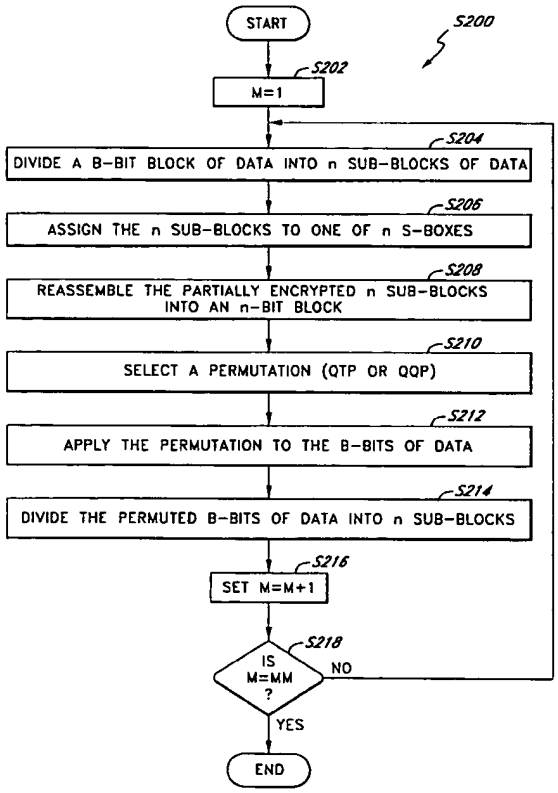
3,796,830	3/1974	Smith .	
3,798,359	3/1974	Feistel .	
3,962,539	6/1976	Ehrsam et al. .	
4,078,152	3/1978	Tuckerman, III .	
4,195,200	3/1980	Feistel .	
4,255,811	3/1981	Adler .	
4,316,055	2/1982	Feistel .	
4,322,577	3/1982	Brändström .	
4,520,232	5/1985	Wilson .	
4,668,103	5/1987	Wilson .	
4,685,132	8/1987	Bishop et al. .	
4,731,843	3/1988	Holmquist	380/29
4,751,733	6/1988	Delayaye et al. .	
4,797,921	1/1989	Shiraishi .	
4,932,056	6/1990	Shamir .	
4,979,832	12/1990	Ritter .	
5,003,596	3/1991	Wood .	
5,008,935	4/1991	Roberts	380/29
5,038,376	8/1991	Mittenthal .	
5,214,704	5/1993	Mittenthal .	
5,245,658	9/1993	Bush et al. .	
5,297,206	3/1994	Orton .	
5,317,639	5/1994	Mittenthal .	
5,511,123	4/1996	Adams .	
5,623,548	4/1997	Akiyama et al. .	
5,623,549	4/1997	Ritter .	

Primary Examiner—Thomas H. Tarcza
Assistant Examiner—Pinchus M. Laufer
Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

[57] **ABSTRACT**

A method and apparatus for inter-round mixing in iterated block substitution systems is disclosed. The method involves optimizing inter-round mixing so that each data bit affects each other data bit in the same way. This is accomplished by applying a quick trickle permutation or a quasi quick trickle permutation to the data bits undergoing block substitution allocated to n individual substitution boxes.

6 Claims, 21 Drawing Sheets



US005838795A

United States Patent [19]

Mittenthal

[11] Patent Number: 5,838,795

[45] Date of Patent: Nov. 17, 1998

[54] **METHOD AND APPARATUS FOR STATISTICAL DIFFUSION IN ITERATED BLOCK SUBSTITUTION**

[75] Inventor: **Lothrop Mittenthal**, Thousand Oaks, Calif.

[73] Assignee: **Teledyne Industries, Inc.**, Los Angeles, Calif.

[21] Appl. No.: **888,454**

[22] Filed: **Jul. 7, 1997**

Related U.S. Application Data

[62] Division of Ser. No. 584,523, Jan. 11, 1996.

[51] **Int. Cl.**[6] **H04L 9/28**

[52] **U.S. Cl.** **380/28**; 380/37; 380/42

[58] **Field of Search** 380/28, 29, 37, 380/42, 59

[56] **References Cited**

U.S. PATENT DOCUMENTS

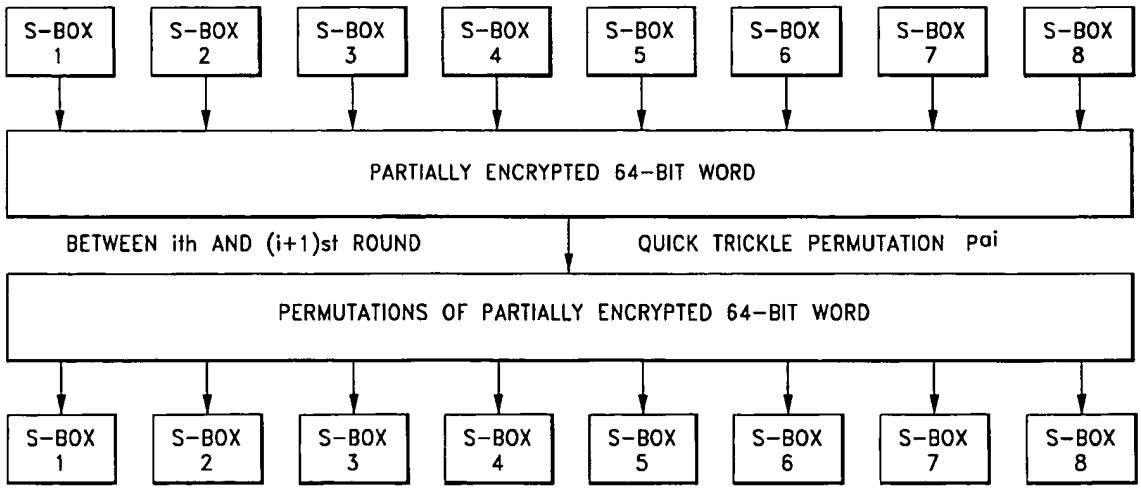
3,796,830	3/1974	Smith .	
3,798,359	3/1974	Feistel .	
3,962,539	6/1976	Ehrsam et al. .	
4,078,152	3/1978	Tuckerman, III .	
4,195,200	3/1980	Feistel .	
4,255,811	3/1981	Adler .	
4,316,055	2/1982	Feistel .	
4,322,577	3/1982	Brändström .	
4,520,232	5/1985	Wilson .	
4,668,103	5/1987	Wilson .	
4,685,132	8/1987	Bishop et al. .	
4,751,733	6/1988	Delayaye et al. .	
4,797,921	1/1989	Shiraishi .	
4,932,056	6/1990	Shamir .	
4,979,832	12/1990	Ritter .	
5,003,596	3/1991	Wood .	
5,038,376	8/1991	Mittenthal .	
5,214,704	5/1993	Mittenthal .	
5,245,658	9/1993	Bush et al. .	
5,297,206	3/1994	Orton .	
5,317,639	5/1994	Mittenthal .	
5,511,123	4/1996	Adams .	
5,623,548	4/1997	Akiyama et al. .	
5,623,549	4/1997	Ritter .	

Primary Examiner—Thomas H. Tarcza
Assistant Examiner—Pinchus M. Laufer
Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

[57] **ABSTRACT**

A method and apparatus for inter-round mixing in iterated block substitution systems is disclosed. The method involves optimizing inter-round mixing so that each sub-block of data affects each other in the same way. This is accomplished by applying a quick trickle permutation or a quasi quick trickle permutation to blocks of data allocated to n individual substitution boxes.

9 Claims, 21 Drawing Sheets



US005838794A

United States Patent [19]

Mittenthal

[11] Patent Number: 5,838,794

[45] Date of Patent: Nov. 17, 1998

[54] **METHOD AND APPARATUS FOR INTER-ROUND MIXING IN ITERATED BLOCK SUBSTITUTION SYSTEMS**

[75] Inventor: **Lothrop Mittenthal**, Thousand Oaks, Calif.

[73] Assignee: **Teledyne Electronic Technologies**, Newbury Park, Calif.

[21] Appl. No.: **584,523**

[22] Filed: **Jan. 11, 1996**

[51] **Int. Cl.**[6] **H04L 9/28**

[52] **U.S. Cl.** **380/28**; 380/29; 380/37; 380/42

[58] **Field of Search** 380/37, 42, 28, 380/29

[56] **References Cited**

U.S. PATENT DOCUMENTS

3,796,830	3/1974	Smith	380/37
3,798,359	3/1974	Feistel	380/37
3,962,539	6/1976	Ehrsam et al.	380/29
4,078,152	3/1978	Tuckermann, III	380/37
4,195,200	3/1980	Feistel	380/37
4,255,811	3/1981	Adler	380/37
4,316,055	2/1982	Feistal	380/37
4,322,577	3/1982	Brändström	380/37
4,520,232	5/1985	Wilson	380/28
4,668,103	5/1987	Wilson	380/30
4,685,132	8/1987	Bishop et al.	380/46
4,751,733	6/1988	Delayaye et al.	380/42
4,797,921	1/1989	Shiraishi	380/28
4,932,056	6/1990	Shamir	380/23
4,979,832	12/1990	Ritter	380/28
5,003,596	3/1991	Wood	380/28
5,038,376	8/1991	Mittenthal	380/37
5,214,704	5/1993	Mittenthal	380/37
5,245,658	9/1993	Bush et al.	380/28
5,297,206	3/1994	Orton	380/30
5,317,639	5/1994	Mittenthal	380/37
5,511,123	4/1996	Adams	380/29
5,623,548	4/1997	Akiyama et al.	380/28
5,623,549	4/1997	Ritter	380/37

FOREIGN PATENT DOCUMENTS

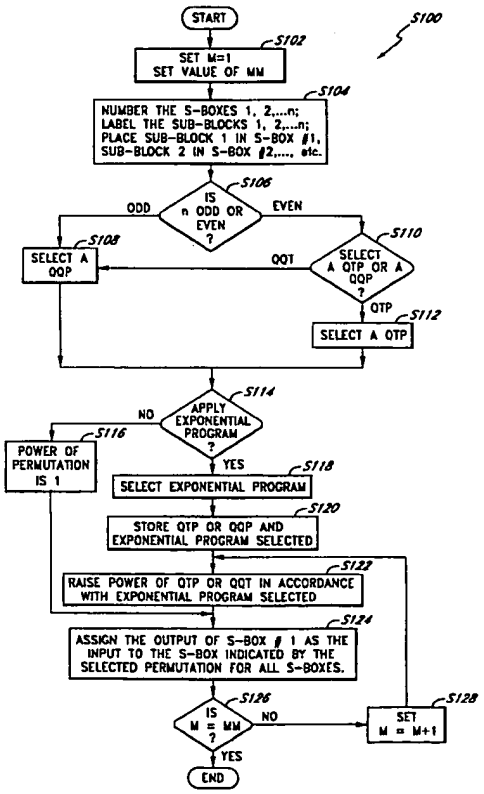
0 406 457	1/1991	European Pat. Off.	H04L 9/06
0 411 712	6/1991	European Pat. Off.	H04L 9/06

Primary Examiner—Thomas H. Tarcza
Assistant Examiner—Pinchus M. Laufer
Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

[57] **ABSTRACT**

A method and apparatus for inter-round mixing in iterated block substitution systems is disclosed. The method involves optimizing inter-round mixing so that each sub-block of data affects each other in the same way. This is accomplished by applying a quick trickle permutation or a quasi quick trickle permutation to blocks of data allocated to n individual substitution boxes.

16 Claims, 21 Drawing Sheets



28/3,K/11 (Item 11 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014058695 **Image available**
WPI Acc No: 2001-542908/200161
XRPX Acc No: N01-403630
Extended key generator, encryption / decryption unit and storage medium where each key transform function carries out transform process by S box on the basis of a key from the inputted key, adder computes corresponding extended key
Patent Assignee: TOSHIBA KK (TOKE); MATSUSHITA ELECTRIC IND CO LTD (MATU); MATSUSHITA DENKI SANGYO KK (MATU); TOSHIBA CORP (TOKE)
Inventor: KAWAMURA S; OOMORI M; SANO F; SEKIBE T; TATEBAYASHI M; YOKOTA K; KKAWAMURA S; SKIBE T
Number of Countries: 032 Number of Patents: 011
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 1081889	A2	20010307	EP 2000307086	A	20000818	200161	B
CN 1291744	A	20010418	CN 2000126869	A	20000831	200161	
JP 2001142395	A	20010525	JP 2000261098	A	20000830	200161	
KR 2001067121	A	20010712	KR 200050093	A	20000828	200202	
SG 92735	A1	20021119	SG 20004785	A	20000821	200303	
JP 3389210	B2	20030324	JP 2000261098	A	20000830	200323	
KR 402811	B	20031030	KR 200050093	A	20000828	200417	
TW 556111	A	20031001	TW 2000116363	A	20000814	200423	
US 6891950	B1	20050510	US 2000652157	A	20000831	200532	
MX 2000008484	A1	20041101	MX 20008484	A	20000830	200557	
EP 1081889	B1	20051214	EP 2000307086	A	20000818	200602	

Priority Applications (No Type Date): JP 99244176 A 19990831
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 1081889	A2	E	26	H04L-009/06	
Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI					
CN 1291744	A			G06F-007/00	
JP 2001142395	A		17	G09C-001/00	
KR 2001067121	A			G06F-001/00	
SG 92735	A1			H04L-009/06	
JP 3389210	B2		17	G09C-001/00	Previous Publ. patent JP 2001142395
KR 402811	B			G06F-001/00	Previous Publ. patent KR 2001067121
TW 556111	A			G06F-007/00	
US 6891950	B1			H04L-009/00	
MX 2000008484	A1			H04L-009/06	
EP 1081889	B1	E		H04L-009/06	
Designated States (Regional): DE FR GB NL					

Extended key generator, encryption / decryption unit and storage medium where each key transform function carries out transform process by S box on the basis of a key from the inputted key, adder computes corresponding extended key

Abstract (Basic):
... The unit has each transform function (fk) performs a transform process by an **S box** (14) or **substitution** table, onthe basis of the **first** key (KA) obtained from the inputted key. An adder computes a corresponding extended key (K1-K16) on the basis of a value obtained by shifting the transformed result of the **S box** to the left and a **2nd** key (KB) obtained from the inputted key.
... As an extended key generator, **encryption / decryption** unit and

a storage medium where each key transform function carries out transform process by an **S box** .
...

...Improves the **cryptological** robustness...

...drawing shows a block diagram showing the arrangement of an extended key generator in the **encryption / decryption** unit...
...the **S box** (14...

...the **first** key (KA...

...the **second** key (KB
...Title Terms: **ENCRYPTION** ;
...International Patent Class (Main): **H04L-009/00** ...

... **H04L-009/06**
...International Patent Class (Additional): **H04K-001/00** ...

... **H04K-001/04** ...

... **H04K-001/06**
Manual Codes (EPI/S-X): **T01-D01** ...

... **T01-J04B1** ...

... **W01-A05A**

US006891950B1

(12) **United States Patent**
Oomori et al.

(10) **Patent No.: US 6,891,950 B1**
(45) **Date of Patent: May 10, 2005**

(54) **EXTENDED KEY GENERATOR, ENCRYPTION/DECRYPTION UNIT, EXTENDED KEY GENERATION METHOD, AND STORAGE MEDIUM**

(75) Inventors: **Motoji Oomori**, Hirakata (JP); **Kaoru Yokota**, Ashiya (JP); **Tsutomu Sekibe**, Hirakata (JP); **Makoto Tatebayashi**, Takarazuka (JP); **Fumihiko Sano**, Fuchu (JP); **Shinichi Kawamura**, Kodaira (JP)

(73) Assignees: **Kabushiki Kaisha Toshiba**, Kawasaki (JP); **Matsushita Electric Industrial Co., Ltd.**, Kadoma (JP)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 846 days.

(21) Appl. No.: **09/652,157**

(22) Filed: **Aug. 31, 2000**

(30) **Foreign Application Priority Data**

Aug. 31, 1999 (JP) 11-244176

(51) **Int. Cl.**[7] **H04L 9/00**; H04K 1/00; H04K 1/04; H04K 1/06

(52) **U.S. Cl.** **380/44**; 380/29; 380/37

(58) **Field of Search** 380/44, 29, 37; 235/380

(56) **References Cited**

U.S. PATENT DOCUMENTS

3,958,081 A		5/1976	Ehrsam et al.	
4,255,811 A	*	3/1981	Adler	380/37
4,802,217 A	*	1/1989	Michener	380/29
4,850,019 A		7/1989	Shimizu et al.	
5,317,638 A	*	5/1994	Kao et al.	380/29
5,442,705 A	*	8/1995	Miyano	380/29
5,511,123 A		4/1996	Adams	
5,703,952 A	*	12/1997	Taylor	380/44
5,787,179 A	*	7/1998	Ogawa et al.	380/46
5,949,884 A		9/1999	Adams et al.	
6,246,768 B1	*	6/2001	Kim	380/28
6,256,391 B1	*	7/2001	Ishiguro et al.	380/203
6,292,896 B1	*	9/2001	Guski et al.	713/169
6,570,989 B1	*	5/2003	Ohmori et al.	380/42
6,606,385 B1	*	8/2003	Aikawa et al.	380/28
6,683,956 B1	*	1/2004	Aikawa et al.	380/37

FOREIGN PATENT DOCUMENTS

EP	0618 701 A2	10/1994
EP	1 052 611 A1	11/2000
JP	10-116029 A2	5/1998
JP	2000-261098	6/2002
WO	WO99/14889	3/1999
WO	WO99/38143	7/1999

OTHER PUBLICATIONS

Aiello, William et al. "Design of Practical and Provably Good Random Number Generators",.*

Han, Seung–Jo et al. "The Improved Data Encryption Standard (DES) Algorithm", 1996 IEEE, pp. 1310–1314.*

Keliher, Liam et al. "Provable Security of Substitution–Permutation Encryption Netowrks Against Linear Cryptanalysis", 2000 IEEE, pp. 37–42.*

Lim, Young Won. "Efficient 8–Cycle DES Implementation", 2000 IEEE, pp. 175–178.*

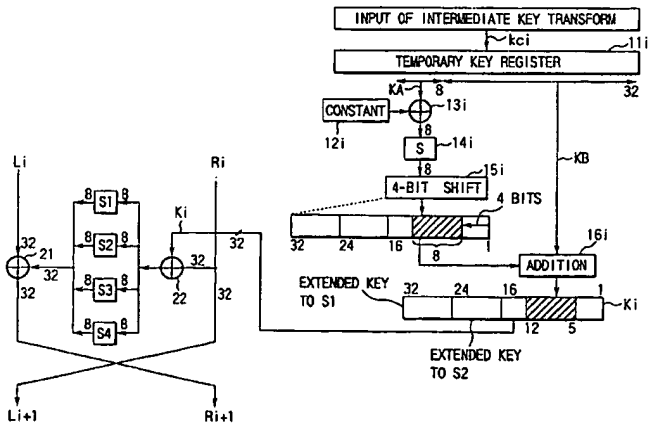
(Continued)

Primary Examiner—Gregory Morse
Assistant Examiner—Michael J. Simitoski
(74) *Attorney, Agent, or Firm*—Finnegan, Henderson, Farabow, Garrett & Dunner, L.L.P.

(57) **ABSTRACT**

There are disclosed an extended key generator, encryption/decryption unit, and storage medium, in which as each of key transform functions, a transform process is done by an S box (substitution table) on the basis of a first key obtained from the inputted key, and an adder computes a corresponding one of extended keys on the basis of a value obtained by shifting the transformed result of the S box to the left, and a second key obtained from the inputted key.

4 Claims, 15 Drawing Sheets



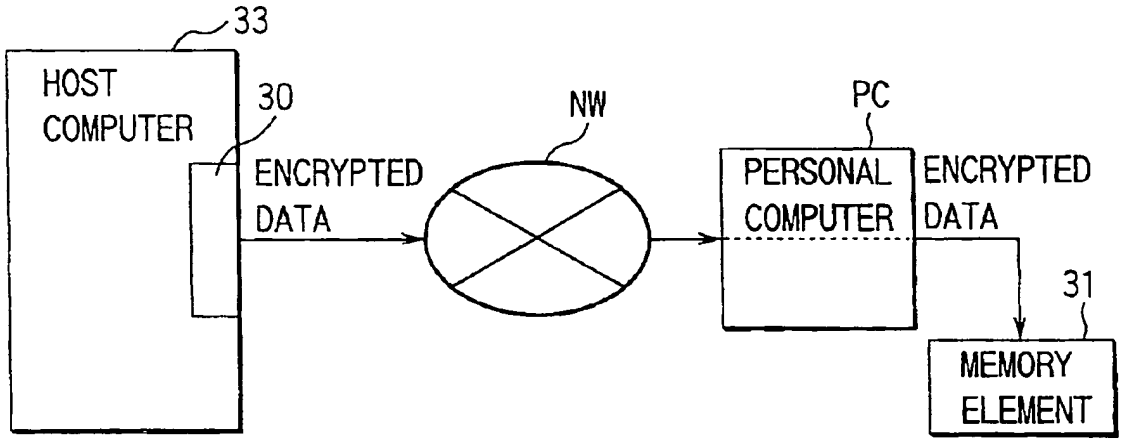


FIG. 17

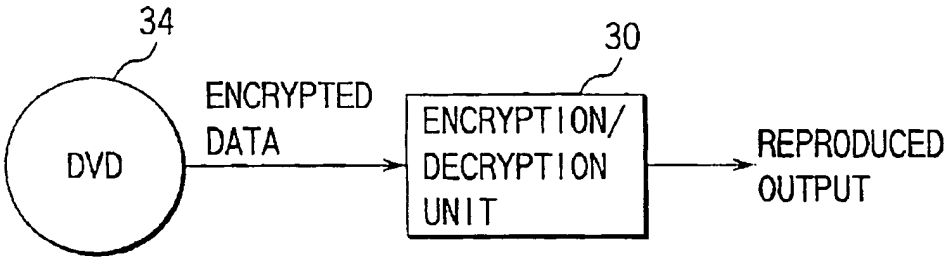


FIG. 18A

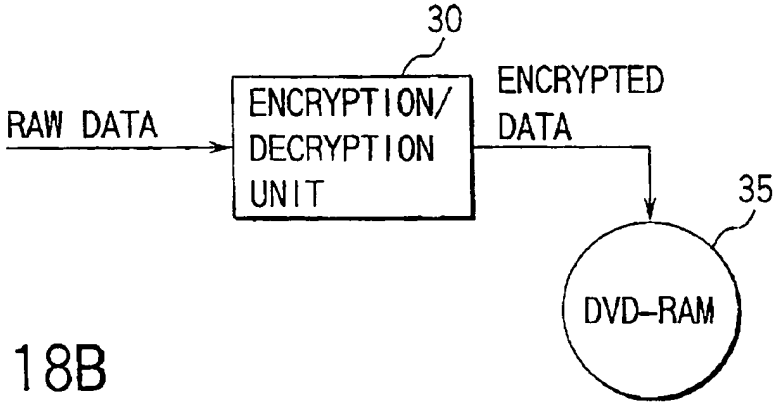


FIG. 18B

14_i, thus similarly practicing the present invention and obtaining the same effect. More specifically, the XORs of the value KA and constants may be computed in advance and are held in the form of a table, and the S box $14x_i$ may look up the table using the value KA as an input parameter to obtain a given XOR.

FIG. **14** is a functional block diagram showing the arrangement of a smart card that embodies the aforementioned extended key generator, encryption/decryption unit, and storage medium of the present invention. As shown in FIG. **14**, a smart card **51** has a CPU **53**, RAM **55**, ROM **57**, EEPROM **59**, and contactor **61**. The RAM **55** is used to store various data, and is used as a work area or the like. The ROM **57** is used to store various data, programs, and the like. The EEPROM **59** stores programs and the like shown in the flow charts in FIGS. **8** and **13**. The contactor **61** obtains electrical contacts with a smart card reader/writer (not shown). Note that the programs shown in FIGS. **8** and **13** may be stored in the RAM **55** or ROM **57** in place of the EEPROM **59**.

Fourth Embodiment

An encryption/decryption unit according to the fourth embodiment of the present invention will be described below using FIG. **15**. This encryption/decryption unit **30** has an arrangement described in one of the first to third embodiments, and is used to protect digital information such as image data, music data, and the like (to be referred to as raw data hereinafter).

Assume that the encryption/decryption unit **30** is implemented on a personal computer PC by installing a program from a storage medium, as shown in FIG. **15**. The encryption/decryption unit **30** encrypts raw data input to the personal computer PC using, e.g., a user ID as a common key, and stores the obtained encrypted data (corresponding to the aforementioned ciphertext) in a portable memory element **31**. AS such memory element **31**, a smart card, smart media, memory card, or the like may be used.

The memory element **31** is distributed to the user's home, and an encryption/decryption unit (not shown) in the user's home decrypts the encrypted data in the memory element **13** on the basis of the self user ID and reproduces obtained image data or music data from, e.g., a loudspeaker or the like. In this manner, raw data (contents) can be distributed to only users who have made a subscription contract in advance.

Various modifications of this embodiment are available as follows. For example, as shown in FIG. **16**, a recording unit **32** comprising the encryption/decryption unit **30** as a hardware circuit may be provided in place of the personal computer PC. With this arrangement, upon writing contents in the memory element **31**, the encryption/decryption unit **30** encrypts raw data based on, e.g., a user ID, and stores encrypted data in the memory element **31**. The processes from delivery to the home to decryption are the same as those described above. In this manner, the encryption/decryption unit **30** may be provided to the dedicated recording unit **32** in place of a versatile computer such as the personal computer PC and the like.

Also, as shown in FIG. **17**, a host computer **33** with the encryption/decryption unit **30** may be connected to the personal computer PC via a network NW. In this case, encrypted data downloaded from the host computer **33** is stored in the memory element **32** via the personal computer PC in the encrypted state. The processes from delivery to the home to decryption are the same as those described above. According to this modification, in addition to the aforementioned effect, contents (raw data) on the network NW can be prevented from eavesdropped.

Furthermore, as shown in FIGS. **18A** and **18B**, a DVD (digital versatile disc) may be used as the memory element. In the case shown in FIG. **18A**, a DVD **34** that pre-stores encrypted data is distributed to the user. The encryption/decryption unit **30** at the user's home decrypts the encrypted data in the DVD **34**, and reproduces obtained image data or music data from a loudspeaker or the like.

Also, in the case shown in FIG. **18B**, raw data such as image data, music data, or the like is encrypted by the encryption/decryption unit **30** at the user's home using a predetermined common key, and the obtained encrypted data is stored in a DVD-RAM **35**.

This encrypted data is decrypted by the predetermined common key set by the user, but cannot be decrypted by a third party unless the common key is disclosed. Therefore, personal image data and music data can be saved while being protected from third parties.

Other Embodiments

As a storage medium that stores a program for implementing the processes of the extended key generator and encryption/decryption unit of the present invention, a magnetic disk, floppy disk, hard disk, optical disk (CD-ROM, CD-R, DVD, or the like), magnetooptical disk (MO or the like), semiconductor memory, and the like may be used. In practice, the storage format is not particularly limited as long as a storage medium can store the program and can be read by a computer.

An OS (operating system) which is running on a computer or MW (middleware) such as database management software, network software, or the like may execute some of processes that implement the above embodiment, on the basis of an instruction of the program installed from the storage medium in the computer.

Furthermore, the storage medium in the present invention is not limited to a medium independent from the computer, but includes a storage medium which stores or temporarily stores a program downloaded from a LAN, the Internet, or the like.

The number of storage media is not limited to one, and the storage medium of the present invention includes a case wherein the processes of the above embodiment are implemented from a plurality of media, and either medium arrangement may be used.

Note that the computer in the present invention executes processes of the above embodiment on the basis of programs stored in the storage medium, and can be either an apparatus consisting of a single device such as a personal computer, or a system built by connecting a plurality of devices via a network.

The computer in the present invention is not limited to a personal computer, and includes an arithmetic processing device, microcomputer, and the like included in an information processing apparatus, i.e., includes all devices and apparatuses that can implement the functions of the present invention via programs.

The present invention is not limited to a DES cryptosystem but can be applied to any other block cryptosystems using round functions. For example, the present invention may be applied to cryptosystems such as Lucifer, LOKI, MISTY1, MISTY2, and SAFER (Secure and Fast Encryption Routine), and the like.

In the above embodiments, the S box makes nonlinear transformation using a substitution table. Alternatively, the S box may make nonlinear transformation using a wiring pattern.

In the embodiment shown in FIG. **10**, two sets of transform elements including the constant registers 12_i, XOR elements 13_i, S boxes 14_i, and extended transformers 15_i are parallelly arranged. Alternatively, three or more sets of transform elements may be parallelly arranged.

Various other modifications of the present invention may be made within the scope of the invention.

Additional advantages and modifications will readily occur to those skilled in the art. Therefore, the invention in its broader aspects is not limited to the specific details and representative embodiments shown and described herein. Accordingly, various modifications may be made without departing from the spirit or scope of the general inventive concept as defined by the appended claims and their equivalents.

What is claimed is:

1. An expansion key generation apparatus, which generates expansion keys based on input keys, the apparatus comprising a plurality of cascade-connected key transform devices, each of the key transform devices comprising:

an exclusive-OR element for calculating an exclusive-OR of a constant determined for each of the key transform devices and a first key obtained from the input key;

a nonlinear transform unit for nonlinearly transforming an output from the exclusive-OR element using a predetermined substitution table;

an expansion unit for performing an expansion processing on an output from the nonlinear transform unit; and

an expansion key calculation unit for calculating the expansion key based on an output from the expansion unit and a second key obtained from the input key,

wherein the expansion key calculation unit adds with carry-up the output from the expansion unit and the second key.

2. An expansion key generation apparatus, which generates expansion keys based on input keys, the apparatus comprising a plurality of cascade-connected key transform devices, each of the key transform devices comprising:

an exclusive-OR element for calculating an exclusive-OR of a constant determined for each of the key transform devices and a first key obtained from the input key;

a nonlinear transform unit for nonlinearly transforming an output from the exclusive-OR element using a predetermined substitution table;

an expansion unit for performing an expansion processing on an output from the nonlinear transform unit; and

an expansion key calculation unit for calculating the expansion key based on an output from the expansion unit and a second key obtained from the input key, wherein the expansion key calculation unit performs a shifting of a predetermined number of bits and shifts the output from the nonlinear transform unit to the least significant bit by the number of bits that is half the number of bits of the output from the nonlinear transform unit, or by the number of bits obtained by adding an integer multiple of the number of bits of the output from the nonlinear transform unit to the half number of bits.

3. An expansion key generation program, which causes a computer to generate expansion keys based on input keys using a plurality of cascade-connected key transform devices, the program comprising:

program code for calculating an exclusive-OR of a constant determined for each of the key transform devices and a first key obtained from the input key;

program code for nonlinearly transforming a result of an exclusive-OR using a predetermined substitution table;

program code for performing an expansion processing on a result of a nonlinear transform; and

program code for calculating the expansion key based on a result of expansion processing and a second key obtained from the input key,

wherein the program code for calculating the expansion key comprises program code for adding with carry-up a result of an expansion and the second key.

4. An expansion key generation program, which causes a computer to generate expansion keys based on input keys using a plurality of cascade-connected key transform devices, the program comprising:

program code for calculating an exclusive-OR of a constant determined for each of the key transform devices and a first key obtained from the input key;

program code for nonlinearly transforming a result of an exclusive-OR using a predetermined substitution table;

program code for performing an expansion processing on a result of a nonlinear transform, wherein the program code for performing the expansion processing comprises program code for shifting a result of a nonlinear transform by a predetermined number of bits that is half the number of bits of a result of a nonlinear transform, or by the number of bits obtained by adding an integer multiple of the number of bits of the result of the nonlinear transform to the half number of bits;

program code for calculating the expansion key based on a result of expansion processing and a second key obtained from the input key; and

program code for shifting the input key to a least significant bit or a most significant bit and inputting the shifted key to the key transform device of a next stage.

* * * * *

28/3,K/29 (Item 29 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014094447 **Image available**
WPI Acc No: 2001-578661/200165
XRPX Acc No: N01-430577
Encryption **program used for electronic mail security, includes instructions for performing mixing of data** segments , **swapping and** substitution **iteratively for preset times using** different **sub-keys**
Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: COPPERSMITH D; GENNARO R; HALEVI S; JUTLA C S; MATYAS S M; PEYRAVIAN M; SAFFORD D R; ZUNIC N
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 6243470	B1	20010605	US 9818707	A	19980204	200165	B

Priority Applications (No Type Date): US 9818707 A 19980204
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6243470	B1		29	H04L-009/06	

Encryption **program used for electronic mail security, includes instructions for performing mixing of data** segments , **swapping and** substitution **iteratively for preset times using** different **sub-keys**

Abstract (Basic):
... The programs include instruction to receive input data with each data **segments** having bytes equal to block length of variable length block for ciphering. The input data **segments** is mixed using XOR operations and **substitution box** (**S - box**) look-up operation. The mixed **segments** are swapped and XOR of swapped **segments** and **S - box** look-up are performed to produce **substituted** bytes. The process is iterated for preset times using **different** sub-keys.
... The sub-keys are generated using the symmetric input key distinctly for every round of **encryption** . INDEPENDENT CLAIMS are also included for the following...

...a) **Encryption** system...

...b) **Encryption** method...

...For **encrypting** input data using block cipher algorithm for secure storage of e.g. customer accounts in...

...The block cipher algorithm allows variation of block size, key size and number of **encryption** cycles and uses logical XOR operation which reduces time used for **encrypting** and **decrypting** data...

...The figure shows the flowchart of **encryption** process...
Title Terms: **ENCRYPTION** ;
International Patent Class (Main): **H04L-009/06**
Manual Codes (EPI/S-X): **T01-D01** ...

... **T01-H01C2** ...

... **T01-H07C1** ...

... **T01-H07C5E** ...

... T01-J05A1 ...

... T01-J05A2 ...

... T01-J12C ...

... W01-A05A

US 20020027987A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2002/0027987 A1

Roelse (43) Pub. Date: Mar. 7, 2002

(54) SUBSTITUTION-BOX FOR SYMMETRIC-KEY CIPHERS

(76) Inventor: Petrus Lambertus Adriaanus Roelse, Eindhoven (NL)

Correspondence Address:
U.S. Philips Corporation
580 White Plains Road
Tarrytown, NY 10591 (US)

(21) Appl. No.: 09/896,197

(22) Filed: Jun. 29, 2001

(30) Foreign Application Priority Data

Jul. 4, 2000 (EP) 00202326.5

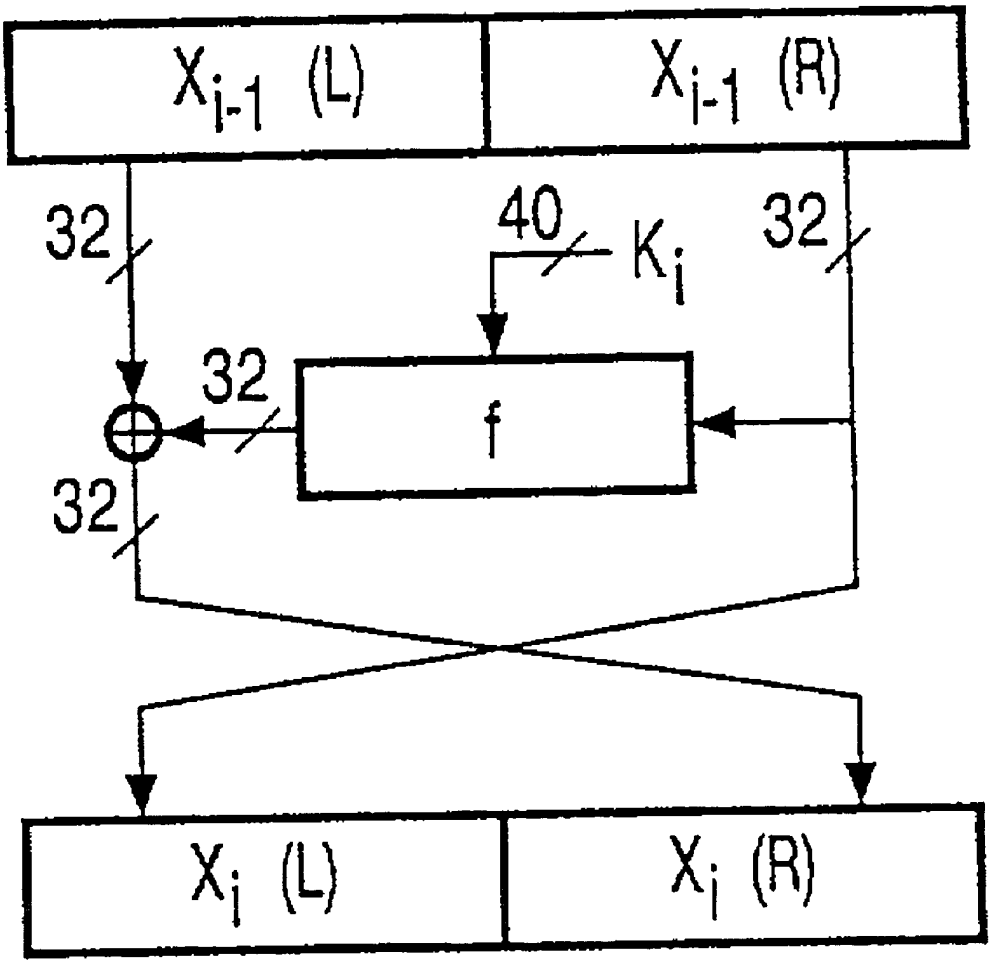
Publication Classification

(51) Int. Cl.[7] H04L 9/06

(52) U.S. Cl. 380/29; 380/37

(57) ABSTRACT

An input data block is cryptographically converted into an output data block; by performing a non-linear operation on the input data block using an S-box based on permutations. The S-box is associated with a set of at least two permutations. Each time before the S-box is used, one of the permutations is (pseudo-)randomly selected from the set of permutations and used for the conversion.



[0057] Linear approximation table of P_0

	β															
iα	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	-2	0	2	2	0	2	0	4	2	4	2	2	0	-2	0
2	0	-2	2	4	-2	0	0	-2	-2	0	0	-2	0	-2	2	-4
3	0	0	-2	-2	0	-4	2	-2	-2	2	4	0	-2	-2	0	0
4	0	-2	2	0	0	2	-2	0	0	2	2	4	-4	2	2	0
5	0	4	2	2	-2	-2	0	4	0	0	2	-2	-2	2	0	0
6	0	0	-4	0	-2	-2	-2	2	2	2	-2	2	0	0	0	-4
7	0	2	0	2	4	-2	-4	-2	-2	0	-2	0	-2	0	-2	0
8	0	4	0	0	2	-2	2	2	0	-4	0	0	-2	2	-2	-2
9	0	-2	0	-2	0	-2	-4	2	0	-2	0	-2	0	-2	4	2
A	0	2	-2	0	4	2	2	0	2	0	0	-2	-2	0	4	-2
B	0	0	2	-2	2	-2	0	0	-2	2	0	0	4	4	2	-2
C	0	-2	-2	0	-2	0	0	-2	0	2	-2	-4	-2	4	0	2
D	0	0	-2	-2	0	4	-2	2	-4	0	2	-2	0	0	-2	-2
E	0	0	-4	4	0	0	0	0	-2	-2	2	2	2	2	2	2
F	0	-2	0	2	2	0	2	4	-2	4	-2	0	0	-2	0	2

[0058] The probability for a given (local) linear characteristic, i.e. the probability that the linear relation on the input bits defined by a equals the linear relation on the output bits defined by β (denoted by $\alpha \rightarrow \beta$), equals $\frac{1}{2}+L_1^{\alpha,\beta}/16$. Note that the entries in the first row and column of these tables represent the trivial characteristic, i.e. $0 \rightarrow 0$ with probability one, which holds for any mapping. It is easily seen that all other (non-trivial) differential characteristics have probability between ¼ and ¾, since the minimum and maximum value over all other entries equal minus four and four respectively for both permutations.

[0059] Linear approximation table of P_1

	β															
α	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	2	0	-2	2	0	2	-4	0	-2	0	2	-2	0	-2	-4
2	0	-2	-2	0	2	-4	4	2	2	0	0	2	0	2	2	0
3	0	-4	2	-2	-4	0	2	-2	-2	-2	0	0	2	2	0	0
4	0	-2	2	4	0	2	2	0	4	-2	2	0	0	-2	-2	0
5	0	0	2	2	-2	-2	0	0	0	0	-2	-2	-2	-2	4	-4
6	0	0	0	0	2	2	-2	-2	2	-2	-2	2	4	0	4	0
7	0	-2	-4	-2	0	2	0	-2	2	0	2	-4	-2	0	2	0
8	0	0	0	4	2	2	2	-2	-4	0	0	0	-2	2	2	2
9	0	-2	0	-2	0	2	0	2	0	-2	-4	2	-4	-2	0	2
A	0	-2	2	0	0	2	-2	0	2	4	0	2	-2	4	0	-2
B	0	0	-2	2	-2	-2	0	-4	2	2	-4	0	0	0	-2	2
C	0	2	-2	0	-2	4	4	2	0	2	-2	0	2	0	0	-2
D	0	0	-2	2	0	0	-2	2	0	-4	-2	-2	0	4	-2	-2
B	0	4	0	0	-4	0	0	0	2	-2	2	2	-2	2	2	2
F	0	-2	-4	2	-2	0	-2	0	-2	0	2	4	0	-2	0	-2

[0060] The compensation effect can, for instance, be seen by considering the linear characteristic $2 \rightarrow 3$ for both permutations. For P_0 the probability that $2 \rightarrow 3$ equals $\frac{1}{2}+L_0^{2,3}/16=\frac{3}{4}$, for P_1 this probability is given by $\frac{1}{2}+L_1^{2,3}/16=\frac{1}{2}$. Preferably this compensation occurs for as many as possible elements. In the example, this holds for all elements with the maximum absolute value of four. Using well-known techniques for generating and testing permutations, a person skilled in the art can create eight such pairs of permutations within a few days for 4-bit permutations. Alternatively, a different pair of permutations P_0^* and P_1^* satisfying the criteria can be constructed from P_0 and P_1 by e.g. applying an affine transformation on the output of both of these permutations. This cane be done by selecting a non-singular 4×4 matrix A over Z_2 and a vector $b \in Z_2^4$ and defining $P_0^*(x) := P_0(x)A \oplus b$ and $P_1^*(x) := P_1(x)A \oplus b$ for all $x \in Z_2^4$ It can be easily verified that in this way 322560 different (ordered) pairs of permutations can be constructed, each of which satisfies all above criteria. Note that one of these transformations is the identity mapping from $Z_2^4 \rightarrow Z_2^4$, i.e. $P_0^*=P_0$ and $P_1^*=P_1$.

1. A method for cryptographically converting an input data block into an output data block; the method including performing a non-linear operation on the input data block using an S-box based on a permutation, wherein the method

includes each time before using the S-box (pseudo-)randomly selecting the permutation from a predetermined set of at least two permutations associated with the S-box.

2. A method as claimed in claim 1, wherein the set of permutations is formed such that a cryptographic weakness in one of the permutations of the set is at least partially compensated by a corresponding cryptographic strength in at least one of the other permutations of the set.

3. A method as claimed in claim 1, wherein the data block consists of n data bits and each element of the set of permutations is a permutation on a set of 2^n elements, represented by Z_2^n, where each non-trivial differential characteristic of each permutation in this set has a probability of at p_{diff}, the set of permutations being formed by permutations which have been selected such that for each non-trivial differential characteristic with probability of p_{diff} in any of the permutations, this differential characteristic has a probability lower than p_{diff} in at least one of the other permutations of the set.

4. A method as claimed in claim 3, wherein the differential characteristic has a probability equal to zero in at least one of the permutations.

5. A method as claimed in claim 4, wherein n=4, and $p_{diff}=¼$.

6. A method as claimed in claim 1, wherein the data block consists of n data bits and each element of the set of permutations is a permutation on a set of 2^n elements, represented by Z_2^n, where each non-trivial linear characteristic of each permutation in this set has a probability of at least $½-p_{lin}$ and at most $½+p_{lin}$, the set of permutations being formed by permutations which have been selected such that for each non-trivial linear characteristic with probability of $½-p_{lin}$ or $½+p_{lin}$ in any of the permutations, this linear characteristic has a probability closer to ½ in at least one of the other permutations of the set.

7. A method as claimed in claim 5, wherein the linear characteristic has a probability equal to ½ in at least one of the permutations.

8. A method as claimed in claim 6, wherein n=4 and $p_{lin}=¼$.

9. A method as claimed in claim 1, wherein the set of permutations consists of two permutations.

10. A method as claimed in claim 1, including performing the selection of the permutation under control of an encryption key.

11. A method as claimed in claim 9 and 10, wherein the selection of the permutation is performed under control of one bit of the encryption key.

12. A computer program product where the program product is operative to cause a processor to perform the method of claim 1.

13. A system for cryptographically converting an input data block into an output data block; the method system including:

an input for receiving the input data block;

a storage for storing a predetermined set of at least two permutations associated with an S-box;

a cryptographic processor for performing a non-linear operation on the input data block using an S-box based on a permutation; the processor being operative to, each time before using the S-box, (pseudo-)randomly selecting the permutation from the stored set of permutations associated with the S-box; and

an output for outputting the processed input data block.

* * * * *